الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي و البحث العلمي
Université Abou Bekr Belkaid
Tlemcen Algérie
جامعة أبي بكر بلقايد
تلمسان الجزائر

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE ABOU BERK BELKAID – TLEMCEN
FACULTE DE TECHNOLOGIE
DEPARTEMENT DE GENIE ELECTRIQUE ET ELECTRONIQUE

Projet de Fin d'Etudes

Filière : Génie Industriel

Intitulé :

Résolution d'un problème d'ordonnancement de type job shop avec contrainte de transport

Réalisé par :

HANNACHE Aboubakr	Ingénierie Des Systèmes
LEMMOUIS Abdelhamid	Chaine logistique

Devant le Jury :

Président	Mr. BENSMAINE Abderrahmane	MCB
Encadreur	Mr. HADRI Abdelkader	MAA
Co-encadreur	Mme KHEDIM Ammaria	MAA
Co-encadreur	Mr. BELBACHIR Driss	Doctorant
Examinateur	Mr. BELKAID Fayçal	MCB
Examinateur	Mr. MEKADER Amin	Ingénieur

REMERCIEMENTS

Nous tenons tout d'abord à remercier fortement Mr HADRI.A et Mme. KHEDIM.A notre tuteurs de projet de fin d'étude, qui nous ont suivi tout au long de cette période et nous ont conseillé sur l'orientation à prendre.

Nous tenons à remercier Mr BELBACHIR.D pour ses précieux conseils, ses commentaires et ses encouragements tout au long de ce travail.

Nous remercions également toute l'équipe de Laboratoire Génie Industriel de l'Université de Tlemcen.

Enfin, nous tenons à remercier tous les enseignants pour leurs aides et le temps qu'ils ont partagé avec nous.

SOMMAIRE

REMERCIEMENTS 2
Introduction générale 4
CHAPITRE 1 14
L'ORDONNANCEMENT DANS LES SYSTEMES DE PRODUCTION 14
1. Introduction : 14
2. Les systèmes de production 14
2.1 Définition d'un système de production 14
2.3 Classification des systèmes de production 15
2.3.1 Classification selon la nature et le volume des flux physiques 15
2.3.2 Classification selon le mode de pilotage : 16
2.4 La gestion de systèmes de production : 16
3. Généralités sur l'ordonnancement 17
3.1. Définition de l'ordonnancement 17
3.2. Les objectifs de l'ordonnancement: 18
3.3. Les éléments d'un problème d'ordonnancement 18
4. Les ateliers de production 21
4.1. Les ateliers une machine: 22
4.2. Les ateliers machines parallèles : 22
4.3. Les ateliers de type flow-shop 22
4.4. Les ateliers de type job shop: 22
4.5. Les ateliers de type open-shop 23
5. Représentation des problèmes d'ordonnancement 14
5.1. Le diagramme de Gantt : 23
5.2. Graphe Potentiel-Tâches 24
5.3. Method PERT (Program Evaluation and ResearcTask) 25
6. Les méthodes de résolution 25
6.1 Les méthodes exactes 27
6.2 Les méthodes approchées 28
6.2.1 Les heuristiques de construction 29
6.2.2 Les méta-heuristiques 30
7. Problème d'ordonnancement de job shop 30
7.2 Historique du Job-Shop 30
7.3 L'ordonnancement de job shop 32
7.4. Définition formelle de problème d'ordonnancement job shop 32
7.4.2 Job shop classique 32

7.4.2 Job-Shop flexible 33
8. Conclusion : 35
CHAPITRE 2 36
LES PROBLEMES D'ORDONNANCEMENT ET LES METHODES DE RESOLUTION 36
1. INTRODUCTION 37
2. Classification des Problèmes d'ordonnancement 37
3. Complexité : 39
4. Les classes P et NP : 39
5. Méthodes de résolution : 42
5.1 Méthodes exactes : 42
5.2 Méthodes approchées : 43
5.2.1 Les heuristiques 43
5.2.2 Les métaheuristiques : 43
5.3 Les méthodes basées sur une seule solution 46
5.3.1 Le recuit simulé : 46
5.3.2 La Recherche Tabou 48
5.4 Les méthodes basées sur une population des solutions : 50
5.4.1 L'optimisation par colonies de fourmis : 50
5.4.2 Description et algorithme : 50
6. Algorithmes Génétiques : 51
6.1 Principe des Algorithmes Génitiques 52
6.2 Codage 53
6.3 Les opérateurs génétiques 54
6.3.1Le croisement 54
6.3.2 La sélection 55
6.4 Paramètres de dimensionnement 55
6.5 La méthode de résolution de notre problème 56
6.5.1 Les avantages des algorithmes génétiques : 56
6.5.2 Les inconvénients des algorithmes génétiques : 56
7. Conclusion : 56
CHAPITRE 3 57
RESOLUTION DE PROBLEME D'ORDONNANCEMENT DE JOB SHOP AVEC CONTRAINTE DE TRANSPORT 57
1. Introduction 58
2. Formulation du problème 58
2.1 Description du System à étudier 58
2.2 Présentation de problème à étudier 60
2.3 Les contraintes 60
2.4 Hypothèses proposées : 60
2.5 Notation 61

3. Démarche de résolution 61
3.1 Le codage 62
3.1.1 Le codage basé sur les jobs : 62
3.2 Création de Population initiale 63
3.3 Croisement 64
3.3.1 Le croisement de l'ordre des Jobs (JOX) 64
3.3.2 La probabilité de croisement Pc : 64
3.4 Mutation 65
3.4.1 Mutation de position 65
3.4.2 La probabilité de mutation Pm 66
3.5 La sélection 66
3.5.1 La sélection par la roue de fortune (roulette de Wheel) 67
3.6 Evaluation Des Individus 68
3.6.1 Calcule du Makespan 68
4. Condition d'arrêt 72
4.1 Arrêt après un nombre de générations fixé à priori 72
4.2 Une combinaison des deux conditions 72
5. Simulation et résultats 72
5.1 Exemple de petite taille (problème de 4 produits) : 72
5.2 Exemple de moyen taille (problème de 6 produits) 79
5.3 Exemple de moyen taille : 81
a. Problème de quinze produits 81
b. Problème de trente produits : 82
6. Conclusion 84
REFERENCES BIBLIOGRAPHIQUES 90
ANNEXE 94
Résumé 95

Liste des figures

Figure I- 1-Caractéristiques d'une tâche ----------20
Figure I 2-Classification des types d'ateliers ----------21
Figure I 3-atelier job shop simple et hybride ----------24
Figure I 4-Problème de job shop classique composé de 3 jobs et 5 machines----------24
Figure I 5-Représentation d'un système de type Job-shop hybride à trois étages----------3424
Figure I 6-diagramme de Gantt ----------28
Figure I 7- Graphe Potentiel-Tâches d'un ordonnancement ----------32
Figure I 8-schémas d'optimisation ----------33
Figure I 9-Espace de recherche de solutions aux COPs----------34
Figure II 1-L'organigramme du recuit simulé 47
Figure II 2-L'organigramme de la recherche tabou 48
Figure II 3-l'influence de l'expérience sur le choix des fourmis 50
Figure II 4-Algorithme de colonies de fourmis de base : the « Ant System » 51
Figure II 5-Principe général des algorithmes génétiques 53
Figure II 6-Exemple d'opérateurs de mutation. 53

Figure III- 1-le system à étudier (system avec 4 machine 55
Figure III- 2-Organigramme général de l'Algorithme Génétique implémenté 60
Figure III- 3-codage de chromosome 63
Figure III- 4-Le croisement de l'ordre des jobs 64
Figure III- 5-Mutation de position 65
Figure III- 6-sélection par la « roue de fortune » 67
Figure III- 7-Le cas de Conflit 68
Figure III- 8-- Le cas de Pas de Conflit 70
Figure III- 9-La population initiale générée 72
Figure III- 10-La population selon l'ordre de Cmax 74
Figure III- 11-Exemple de la procédure de croisement 75
Figure III- 12-Exemple de la procédure de mutation 75
Figure III- 13-Exemple du calcul de Cmax 76
Figure III- 14-Exemple du résultat de sélection 77
Figure III- 15-Résultat finale après 7 itérations 77
Figure III- 16-Exemple de Diagramme de Gant obtenu par Matlab 78
Figure III- 17-Résultat obtenu par notre algorithme après 20 tests 78
Figure III- 18-diagramme de Gantt (chromosome 1->2->3->4) 79
Figure III- 19-Résultat obtenu par Matlab après 7 itérations 80
Figure III- 20-Diagramme de Gant de 6 produits 81
Figure III- 21-Résultat obtenu avec Matlab après 10 itérations 82
Figure III- 22-Résultat obtenu avec Matlab après 10 itérations 84

Liste des tableaux

Tableau-III- 1-les temps de transport entre les machines..... 5872
Tableau-III- 2routage de jobs..... 73
Tableau-III- 3-Temps opératoires sur les machines pour job..... 74
Tableau-III- 4-Résultat obtenu manuellement pour 24 solutions possibles..... 79
Tableau-III- 5-les machines utilisées pour chaque produit..... 80
Tableau-III- 6-Routage de Produits..... 81
Tableau-III- 7-Résultat obtenu manuellement pour 12solutions possibles..... 81
Tableau-III- 9-Les Temps D'opération Des Produits..... 82
Tableau-III- 10-Routage De Produits..... 82
Tableau-III- 11-Les Temps D'opération Des Produits..... 84

Liste des algorithmes

Algorithme- III- 1Génération de la population initiale..... 64
Algorithme- III- 2-le croisement..... 66
Algorithme- III- 3-Mutation..... 67
Algorithme- III- 4-la Sélection..... 68
Algorithme- III- 5-le Makespan (Cmax)..... 72

INTRODUCTION GENERALE

Introduction générale

La situation actuelle des entreprises manufacturières a connu de grands changements. Ce n'est plus un atout de réussite si on s'intéressait uniquement à *produire,* comme c'était le cas autrefois. L'objectif majeur de toute entreprise actuelle n'est pas seulement la *productivité* mais aussi et surtout, la *compétitivité* [69]. Cette compétitivité passe forcément par la diversification du travail, un outil de production de plus en plus flexible et une gestion de production fiable qui devient de plus en plus complexe [70]. L'amélioration de la gestion de production est alors, à la base de toute tentative de changement [71]. Cette fonction vise en effet à organiser le fonctionnement du système de production, et à mieux gérer ses différentes opérations.

Les problèmes d'ordonnancement d'ateliers constituent sûrement pour les entreprises une des difficultés les plus importantes de leurs systèmes de gestion et de pilotage de la production. En effet, c'est à ce niveau que doivent être prises en compte les caractéristiques réelles multiples et complexes des ateliers. Dans ce type d'ordonnancement, les ressources sont généralement des machines, et chaque travail à ordonnancer concerne un produit ou un lot de produits à fabriquer en respectant les gammes de fabrication.

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

Dans ce mémoire nous allons essayer de résoudre un problème d'ordonnancement d'atelier de type job shop avec contrainte de transport où les produits sont transportés dans un sens unidirectionnel l'objectif est de trouver le meilleur ordonnancement de jobs lié au critère de minimisation de la durée de production (Makespan).

La résolution de ce problème de manière optimale s'avère dans la plupart des cas impossible à cause de son caractère fortement combinatoire. Les méthodes exactes requièrent un effort calculatoire qui croît exponentiellement avec la taille du problème. Alors, des méthodes approchées ont été proposées pour résoudre ce problème en temps raisonnable. Parmi ces méthodes, apparaissent celles dites «Métaheuristiques ».

L'avantage de ces Métaheuristiques se résume dans leur adaptabilité à tous les problèmes, et le bon compromis qu'elles présentent entre le temps de recherche, et la qualité de résolution. Grâce à ces Métaheuristiques, on peut proposer des solutions approchées pour des problèmes difficiles et de plus grande taille qu'il était impossible de traiter auparavant.

Pour la résolution de notre problème d'ordonnancement de job shop avec transport, nous avons utilisé les Algorithmes Génétiques dont nous avons pu développer un algorithme qui nous a permet à résoudre le problème dans des temps très faibles.

Ce mémoire est organisé de la manière suivante :

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

Dans le chapitre 2, nous présentons les formulations des problèmes d'optimisation et ses méthodes de résolution à savoir les approches exactes et les approches approximatives.

Le chapitre 3 nous présentons notre problème d'ordonnancement d'atelier de type Job-Shop avec transport à étudier. Puis nous allons présenter la démarche d'adaptation d'algorithme génétique sur notre problématique ainsi que les résultats obtenus après l'application sur plusieurs exemples.

Ce mémoire se termine par une conclusion dans laquelle nous évaluons les résultats et présentons les perspectives de recherche.

CHAPITRE 1

L'ORDONNANCEMENT DANS LES SYSTEMES DE PRODUCTION

1. Introduction :

L'ordonnancement est un champ d'investigation qui a connu un essor important ces dernières années, tant par les nombreux problèmes identifiés que par l'utilisation et le développement de techniques de résolutions .En effet, l'activité d'ordonnancement apparaît dans des disciplines aussi diverses que l'organisation du travail [44].

L'ordonnancement est un processus de décisions, c'est-à-dire un Enchaînement d'événements permettant d'arriver à un but souhaité. À ce processus est associé un environnement, qui est le milieu dans lequel se déroule le problème d'ordonnancement. Par exemple, dans le domaine de l'industrie, le processus de décision peut être l'organisation de la fabrication d'un produit et le but souhaité est de minimiser le temps total de fabrication. Son environnement est l'usine, le personnel et les machines [44].

Ce chapitre est structuré en quatre partie dans la première nous présentons les systèmes de production et ses caractéristiques, la deuxième partie contient des généralités sur l'ordonnancement et les méthodes de représentation d'un problème d'ordonnancement, dans la troisième partie nous allons présenter les différents types d'ateliers en se focalisant surtout sur le problème d'ordonnancement de type job shop. Dans la dernière partie nous présentons les différentes méthodes de résolution.

2. Les systèmes de production

2.1 Définition d'un système de production

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

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

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

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

2.2 Paramètre d'un système de production

A- Taux de production :

Le taux de production est le nombre de produit fabriques par unité de temps. Si le système de système de production fabrique plusieurs types de produits. Le taux de production est donné sous forme de vecteur. **[2]**

B- Capacité :

C'est le débit de sortie maximal pouvant être raisonnablement atteint compte-tenu des différentes contraintes. C'est le taux de production maximum pour le cas d'un seul type de produit. **[2]**

C- Traitement en cours ou stock d'en cours :

C'est le produit dans ses différents stades d'élaboration dans l'usine, parmi les matières premières jusqu'au produit complétement terminé. **[2]**

D- Temps de cycle :

C'est le temps moyen qu'un produit passe dans système de production.

E- Délai :

Le délai est le temps mis entre la connaissance du besoin du command et la fin de réalisation du produit corresponde.

F- Temps de chargement :

C'est le temps nécessaire pour passe de la fabrication d'un type de produit à un autre sur une unité de production donnée**. [2]**

G- Temps d'exécution

C'est le temps pendant lequel un produit subit une transformation lui conférant une valeur ajoutée**. [2]**

H- Temps de transfert :

C'est le temps dépensé pour le transport des unités de produits entre les stations d'opération. **[2]**

I- Temps d'attente :

C'est le temps, pendant lequel un produit ou un lot de produit reste près d'un poste de travail avent d'autre transféré au poste suivant. **[2]**

J- Flexibilité :

C'est la faculté d'un système à s'adapter réellement aux changements de l'environnement. **[2]**

2.3 Classification des systèmes de production

Deux classifications des systèmes de production sont souvent utilisées dans la littérature. La première classification repose sur la nature et le volume des flux physiques dans le Système, tandis que la deuxième tient en compte le mode de pilotage. [3]

2.3.1 Classification selon la nature et le volume des flux physiques

Généralement les systèmes de production se diffèrent, selon la manière de fabrication des produits, en trois classes :

- **Les systèmes travaillant en continu ou en série**:

La caractéristique principale de ces Systèmes est la circulation des matières en flux continu. Les ressources sont organisées Sous forme d'une chaine où les matières premières doivent passer sur toutes ces ressources. Ce type de systèmes concerne surtout les industries dites de « process » dont la production nécessite la manipulation de matières liquides ou gazeuses. [4]

- **Les systèmes à flux discret :**

Ce modèle de systèmes représente le caractère essentiel des entreprises manufacturières. Les produits peuvent être distingués individuellement et les stocks temporaires entre les postes de travail sont souvent utilisés. Les systèmes à flux discret sont devisés en trois classes : [5]

❖ **les systèmes de production en grande série ou de masse**:

la fabrication dans ce type de système est consacré à la réalisation d'une grande quantité de produits identiques, dont le passage de ces produit est toujours le même et dépend de leurs types.

❖ **les systèmes de production en moyenne série ou par lots:**

Contrairement à la classe Précédente, il s'agit ici d'ateliers dans lesquels la diversité des produits nécessite une variété de moyens de production. Les différents produits suivent leur propre chemin sur des ressources communes et qui sont souvent regroupées par fonctionnalités équivalentes.

❖ **les systèmes de production unitaire:**

Ce type de système se caractérise par la faible quantité des produits à fabriquer. Réunir les moyens nécessaires au bon moment et au bon endroit représente la tâche principale de la production pour ce type de système.

❖ **Les systèmes à flux hybride ou discontinu:**

Ces systèmes se situent entre les deux types de systèmes précédents. Deux configurations peuvent être distinguées :

- **les deux types de systèmes (continu et discret) sont couplés**:

  La production est continue tout en ayant un conditionnement discret des produits. On trouve par exemple les systèmes de production de la semoule.

- **les deux aspects continu et discret cohabitent**:

  Dans le même système de production, les traitements sont continus mais effectués par lots. Exemple: la production des boissons.

2.3.2 Classification selon le mode de pilotage :

Dans cette classification les systèmes de production sont classés, selon le type de gestion et le choix de stratégie de fabrication, en deux classes :

➢ **Les systèmes de production à la commande ;**

La production pour ces systèmes est déclenchée par l'arrivée d'une certaine commande. Ce type représente généralement les entreprises fabriquant une variété de produits dont la demande est aléatoire ou celles qui ne définissent pas leurs produis qu'à partir d'une demande précise de client. [4]

- **Les systèmes à flux poussés ;**

Dans ce cadre, le déclenchement de la production est basé sur des planifications et des prévisions pour déterminer un programme de production. La disponibilité du produit venant de l'amont est le responsable de déclenchement de l'étape suivante de fabrication. Cette méthode de production implique souvent le stockage des produits finis avant leur livraison. [5]

2.4 La gestion de systèmes de production :

- **Définition et rôle de la gestion de production :**

La gestion de production a pour objet la recherche d'une organisation efficace dans l'espace et dans le temps de toutes les activités relatives à la production afin d'atteindre les objectifs de l'entreprise [6]. La gestion de production est une fonction complexe, elle s'occupe d'un ensemble de problèmes liés à la production tels que la gestion des données, la planification, le contrôle (suivi) de production, la gestion des stocks, la prévision, l'ordonnancement etc... [7].

- **Organisation hiérarchique de la gestion de production :**

Généralement, la fonction gestion de production est répartie en trois niveaux hiérarchiques: stratégique, tactique et opérationnel [6, 8].

– **Niveau stratégique** : Contient toutes les décisions de caractère 'long terme'. Ces décisions consistent à déterminer la politique de l'entreprise et conditionner son avenir. Elles portent essentiellement sur la gestion des ressources durables, afin que celles-ci Soient toujours suffisantes pour assurer la pérennité de l'entreprise. Les ressources visées peuvent être des machines, des hommes, des informations ou des données Techniques.

– Niveau tactique : Dans ce niveau, les décisions sont prises à moyen terme. Leur rôle est d'assurer la liaison entre le niveau stratégique et le niveau opérationnel et de contrôler la bonne adéquation des ressources disponibles et des charges engendrées par les commandes ou les prévisions, mais sans modification profonde de la structure et du fonctionnement de l'entreprise.

– Niveau opérationnel : Ce niveau représente la gestion quotidienne de l'entreprise. Il Englobe les décisions prises à court terme et à très court terme. Ces décisions assurent le

Lancement des activités et la flexibilité nécessaire à la bonne conduite de la production. On trouve dans ce niveau par exemple : la gestion des stocks, la gestion de la main D'œuvre et la gestion des équipements [9].

- **Fonction ordonnancement dans la gestion de production :**

La gestion de production s'occupe d'un ensemble de problèmes liés à la production tel que la gestion de données, la planification, l'ordonnancement, la gestion de stocks …
La fonction ordonnancement dans le système de gestion de la production joue un rôle très Important, elle consiste à gérer le temps et le rythme de la vie de l'usine.
L'ordonnancement d'atelier couvre un ensemble d'actions qui transforment les décisions de fabrication définies par le programme directeur de production en instructions d'exécution détaillées destinées à piloter et contrôler à court terme l'activité des postes de travail dans l'atelier. Elle peut être décomposée en trois sous-fonctions [10]:

• Élaboration des ordres de fabrication : cette tâche consiste à transformer les informations du programme directeur de production (suggestions de fabrication) en ordres de fabrication;

• Élaboration du planning d'atelier : cette tâche consiste, en fonction de ces ordres de Fabrication et de la disponibilité des ressources consommables (matières premières, Composants) et partageables (postes de travail), à déterminer le calendrier prévisionnel de fabrication (cela revient à transformer les prévisions de fabrication à court terme en ordres d'exécution à très court terme);

• Lancement et suivi : cette tâche consiste à distribuer aux postes de travail les documents nécessaires à la bonne exécution des fabrications (lancement en fabrication) et suivre l'exécution des fabrications (suivi de production).

3. Généralités sur l'ordonnancement

3.1. Définition de l'ordonnancement

« *Ordonnancer, c'est programmer l'exécution d'une réalisation en attribuant des ressources aux tâches et en fixant leurs dates d'exécution* » **[11].**

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 sont présents dans tous les secteurs d'activités de l'économie, depuis l'industrie manufacturière **[12]** jusqu'à l'informatique **[13].**

Ordonnancer le fonctionnement d'un système industriel de production consiste à gérer l'allocation des ressources au cours du temps, tout en optimisant au mieux un ensemble de critères **[14]**.C'est aussi programmer l'exécution d'une réalisation en attribuant des ressources aux tâches et en fixant leurs dates d'exécution **[15].**

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

D'une manière plus simple, un problème d'ordonnancement consiste à affecter des tâches à des ressources à des instants donnés pour répondre au mieux aux besoin²s exprimés par un client, au meilleur coût et dans les meilleurs délais, tout en tenant compte des contraintes.

Les problèmes d'allocation des ressources, d'organisation des tâches, de respect des délais et de prise de décision en temps requis constituent autant de difficultés qu'il est nécessaire de surmonter dans la gestion des systèmes de production en milieu industriel.

Au niveau de l'entreprise, l'ordonnancement concerne plusieurs départements : les ventes, la production, la maintenance, etc. et aussi jouer un rôle très important, car il permet une gestion de ces différents postes qui peut être optimale.

Pour la bonne gestion de ces postes ainsi que des contraintes pouvant y être reliées, il est nécessaire :

- de déterminer les différentes opérations à exécuter, les dates correspondantes, les ressources (matériels et humains) à y affecter,
- d'exécuter ces opérations et de contrôler les coûts qui en découlent.

C'est ainsi que l'ordonnancement intervient pour permettre la meilleure gestion possible du système de production.

3.2. Les objectifs de l'ordonnancement:

Le traitement de l'ordonnancement dans la littérature s'est tout d'abord orienté vers une optimisation monocritère. L'environnement manufacturier évoluant rapidement et la concurrence devenant de plus en plus acharnée, les objectifs des entreprises se sont diversifiés et le processus d'ordonnancement est devenu de plus en plus multicritère. Les critères que doit

satisfaire un ordonnancement sont variés. D'une manière générale on distingue plusieurs classes d'objectifs concernant un ordonnancement **[16].**

- **Les objectifs liés au temps:**

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

- **Les objectifs liés aux ressources** :

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

- **Les objectifs liés au coût:**

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

- **Les objectifs liés à l'énergie ou au débit**.

Les objectifs à satisfaire au niveau de l'ordonnancement son issus des objectifs globaux de l'entreprise par décomposition. Cette décomposition conduit à une structure d'objectifs qui permet de gérer les contradictions et les compromis **[17] et [18].**

3.3. Les éléments d'un problème d'ordonnancement

L'ordonnancement est décrit généralement par les Quatre éléments sont retenus pour décrire d'une façon explicite un ordonnancement, ces éléments sont : les tâches, les ressources, les contraintes et les critères à prendre en considération lors du processus d'optimisation. Dans ce qui suit, on donnera une définition détaillée de chacun de ces éléments.

- **Les tâches :**

Une tâche (task en l'anglais) fait référence à la réalisation d'une opération, cette dernière étant caractérisée par des informations pertinentes nécessaires à la résolution du problème étudié. Ces informations peuvent être différentes selon des critères (l'objectif de l'étude et le point de vue). A titre d'exemple, le planeur s'intéressera au temps requis pour réaliser l'opération sur une machine donnée, au type de ressources que cette opération consomme et à la quantité, tandis que l'opérateur va lui s'intéresser au procédé de réalisation de l'opération.

Dans le cadre des problèmes d'ordonnancement, parmi les informations utiles, on trouve le temps nécessaire pour la réalisation d'une tâche, la(les) ressource(s) qu'elle requiert (une ressource étant renouvelable ou non), une date de début au plus tôt, une date de fin souhaitée et une date de fin impérative. Des contraintes de précédence conditionnent l'exécution d'une tâche par rapport à la fin d'exécution d'autres tâches. Une tâche est alors considérée comme l'exécution d'une opération sur une certaine machine. Plusieurs tâches peuvent être regroupées pour constituer un « job » pour lequel on donne la gamme de production. Il est également nécessaire de connaitre la nature de la tâche, par exemple savoir si une tâche est interruptible ou non, c'est-à-dire de savoir si le traitement d'une tâche sur une ressource donnée peut s'interrompre pendant une certaine durée et de le poursuivre ensuite, cela se traduit par l'autorisation de la préemption des tâches ou non. Les notations suivantes sont introduites pour décrire une opération. Chaque variable, représentera une information particulière se rapportant à la réalisation de l'opération :

- ri pour représenter la date de disponibilité de la tâche i (release date en anglais) ;
- ti pour représenter la date de début de la tâche i (start date en anglais) ;
- ci pour représenter la date de fin d'exécution de la tâche i (completion time en anglais)
- di pour représenter la date d'échéance de la tâche i (due date en anglais) ;
- pi pour représenter la durée opératoire de la tâche i (processing time date en anglais) ;
- F = ci – ri pour représenter la durée de séjour de l'opération i sur la machine avant qu'elle redevienne disponible (flow time en anglais) ;
- Li = ci−di exprime le retard algébrique (lateness) entre la fin d'exécution de la tâche i par rapport à sa date d'échéance ;
- Ti=max(Li,0) qui exprime le retard absolu de la tâche i (tardiness) ;
- Ei =max(− Li, 0) qui exprime l'avancement (earliness) de la tâche i ;

Ces variables sont notamment retenues dans la littérature pour représenter une opération.

La Figure-I-1- montre les relations entre les différentes variables représentant une tâche.

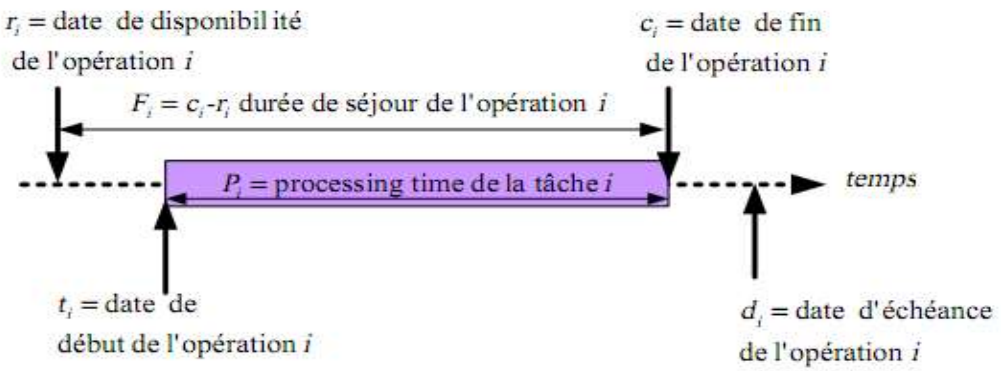


Figure I- 1-Caractéristiques d'une tâche

- **Les ressources :**

Deux familles de ressources existent :

la première famille de ressources est dite renouvelable c'est-à-dire que ces ressources sont réutilisables après la fin d'exécution des opérations. Cette famille est assemblée de machines d'hommes et des outils de productions. Généralement des ressources additionnelles sont considérées comme par exemple l'utilisation des moyens de transport, ou des robots, ou des ressources d'un autre genre comme les buffers d'entrée/sortie. Pendant ces ressources on distingue aussi, des ressources disjonctives qui ne peuvent exécuter qu'une seule opération à la fois et des ressources cumulatives qui savent exécuter plusieurs opérations à la fois par exemple des machines parallèles; notons qu'une machine fait partie de l'ensemble des ressources renouvelable : une machine de capacité 1 est une ressource disponible en un unique exemplaire.

la deuxième famille est dite consommable et regroupe toutes les ressources qui deviennent indisponibles après une utilisation, par exemple des matières premières de l'argent, de l'énergie etc.

- **Les contraintes**

Selon **[16]** les contraintes expriment les restrictions que peuvent prendre conjointement les variables de décision. Donc les contraintes nous renseignent sur les limites imposées par l'environnement. On distingue plusieurs types de contraintes, qui sont **[19], [16]:**

• Les contraintes de positionnement temporelles, Pour la réussite d'un projet ou d'un plan de production, on doit se soumettre aux impératifs de production (réalisation) traduits par le respect d'un planning fixé avant. Au niveau global on parlera d'une date de lancement d'une tâche et d'une date de livraison. A un autre niveau de détail plus fin, pour une tâche ou une

opération. Ce dernier niveau se traduit par la définition des dates suivante : (i) date de disponibilité : ri ; (ii) date d'échéance : di.

- Les contraintes de précédence, une contrainte qui lie le début d'une activité à la fin d'un autre est appelée contrainte de succession ou de précédence. Les gammes opératoires sont un exemple de ces contraintes, d'autres contraintes sont imposées dans certain cas telles que les contraintes de synchronisation, de simultanéité, ou de recouvrements etc. **[16]**.
- les contraintes de ressources, Les contraintes de ressources représentent le fait que les activités requièrent tout au long de leur exécution une certaine quantité de ressources **[19].**Les contraintes de ressources concernent les deux points suivants: (i) l'utilisation de ces ressources, (leur nature, la quantité nécessaire, et les caractéristiques d'utilisation) ; (ii) la disponibilité et la quantité de ces ressources. **[16]**

Ces contraintes nous donnent une information précise sur la nature de l'atelier, et influencent le choix des méthodes d'optimisation ;

Il est important de noter qu'en fonction des objectifs de l'étude, ces contraintes peuvent être strictes ou non. Selon **[20]**, lorsqu'elles sont strictes, elles constituent des obligations à respecter. Lorsqu'elles ne sont pas strictes, on peut ne pas les satisfaire et on les appelle alors des contraintes de préférence.

4. Les ateliers de production

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

Comme le montre la figure1.2, On distingue les trois types d'ateliers suivants : flow-shop, job-shop et open-shop, avec des extensions possibles pour chacun d'eux.

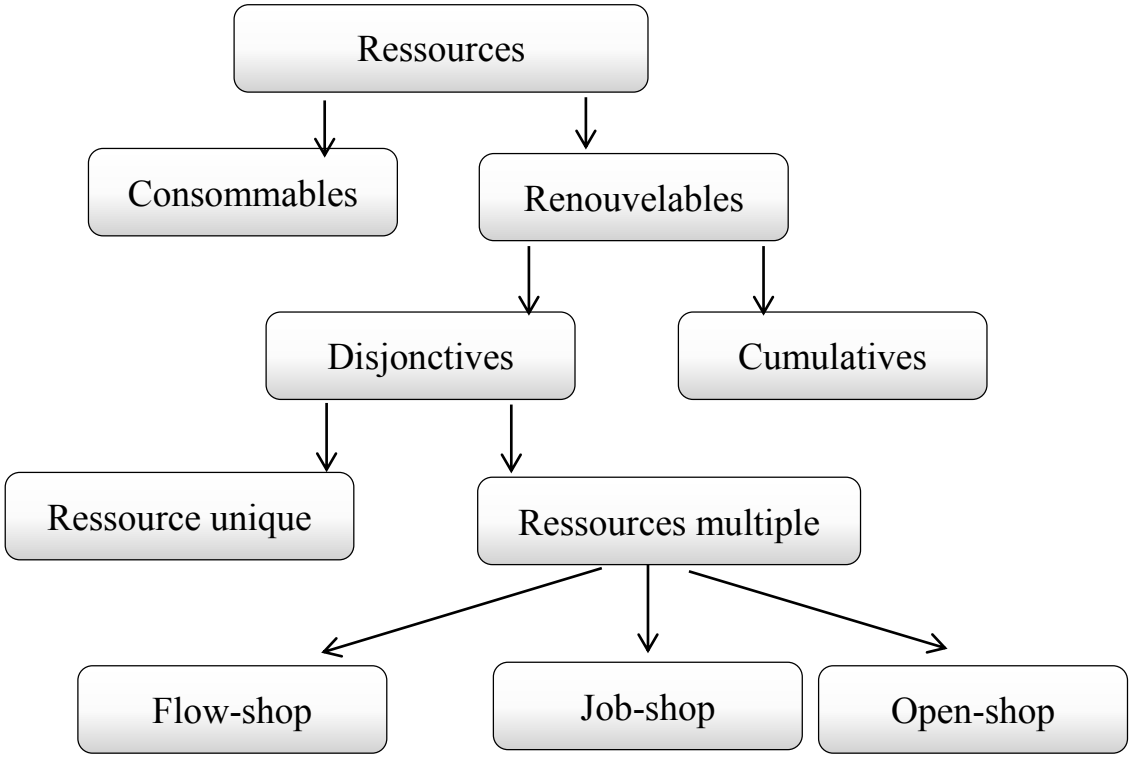


Figure I 2-Classification des types d'ateliers

4.1. Les ateliers une machine:

Chaque job est constitué d'une seule opération et chaque opération peut être réalisée par une seule machine.

4.2. Les ateliers machines parallèles :

Chaque job est constitué d'une seule opération et chaque opération peut être réalisée par n'importe laquelle des machines, disposées en parallèle, mais n'en nécessite qu'une seule.

4.3. Les ateliers de type flow-shop

Appelés également ateliers à cheminement unique, ce sont considèrent comme une ligne de fabrication qui constituée de plusieurs machines en série; toutes les opérations de toutes les tâches passent par les machines dans le même ordre. Dans les ateliers de type flow-shop hybride, une machine peut exister en plusieurs exemplaires identiques fonctionnant en parallèle.

4.4. Les ateliers de type job shop:

Appelés également ateliers à cheminement multiple; Sur cette type l'ordre de passage des jobs sur les machines peut être défirent d'un job à l'autre. Il y a des possibles de passage d'un

job sur un machin plusieurs fois, alors la chaine de ces jobs sur les ateliers de job shop est une boucle.

4.5. Les ateliers de type open-shop

Ces ateliers généralement est moins contraint que celui de type flow-shop ou de type job-shop. Ainsi, l'ordre des opérations n'est 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. Comparé aux autres modèles d'ateliers, l'open shop n'est utilisé pas facilement dans les entreprises.

5. Représentation des problèmes d'ordonnancement

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

5.1. Le diagramme de Gantt :

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

Donc, sur ce diagramme sont indiqués, selon une échelle temporelle: l'occupation des machines par les différentes tâches, les temps morts et les éventuelles indisponibilités des machines dues aux changements entre produits **[38]**

Deux types de diagramme de Gantt sont utilisés : Gantt ressources et Gantt tâches (voir la Figure I -3)**[16]:**

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

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

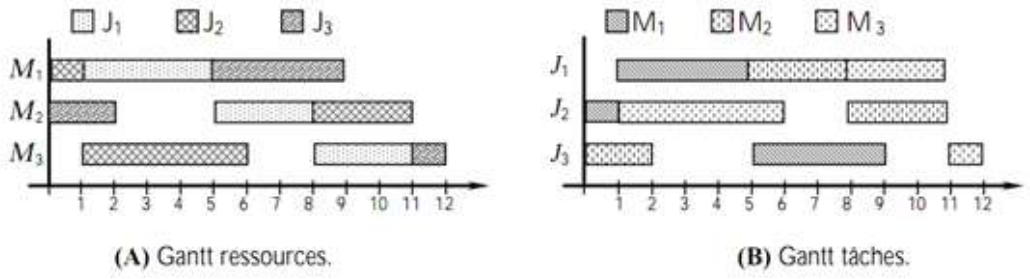


Figure I -3-diagramme de Gantt

5.2. Graphe Potentiel-Tâches

Cette outil graphique a été développé grâce à la théorie des réseaux de Pétri qui ont surtout servi à modéliser les systèmes dynamiques à évènements discrets [**39**].

Dans ce genre de modélisation, les tâches sont représentées par des nœuds et les contraintes par des arcs **[40]**, comme le montre la (Figure I- 4-).

Ainsi, les arcs peuvent être de deux types :

- les arcs conjonctifs illustrant les contraintes de précédence et indiquant les durées des tâches,
- les arcs disjonctifs indiquant les contraintes de ressources **[41], [42]**.

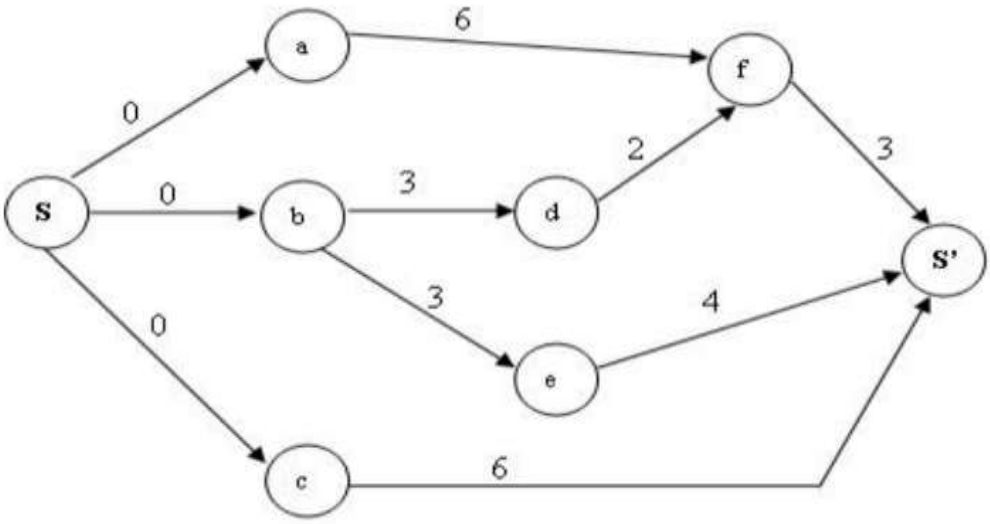


Figure I- 4- Graphe Potentiel-Tâches d'un ordonnancement

Dans l'exemple représenté dans Figure I- 4- la tâche « f » ne peut s'exécuter que si a et d ont été réalisées. Donc, pour exécuter « a » il faut 6 mois et pour exécuter « d » il faut 2+3

mois. La tâche « f » ne pourra commencer au plus tôt que 6 mois après le début du projet: c'est donc le plus long chemin entre « a » et « f ».

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

5.3. Method PERT (Program Evaluation and ResearcTask)

Cette représentation, semblable à la précédente, permet de représenter une tâche par un arc, auquel est associé un chiffre qui représente la durée de la tâche. Entre les arcs, figurent des cercles, appelés sommets ou événements, qui marquent l'aboutissement d'une ou de plusieurs tâches. Ces cercles sont numérotés afin de suivre l'ordre de succession des divers évènements.

Les méthodes graphiques ont connu une très importante évolution surtout avec l'apparition des Réseaux de Pétri (RdP) **[43], [39]**, qui permettent de traduire plusieurs notions fondamentales ayant un lien avec les problèmes d'ordonnancement, telles que :

- les conflits sur les ressources,
- les durées opératoires certaines,
- les durées opératoires aléatoires,
- les gammes,
- les disponibilités, les multiplicités et les capacités de ressources
- la répétitivité, etc …

6. Les méthodes de résolution

On peut séparer les méthodes d'optimisation en deux familles distinctes : les méthodes exactes et les méthodes approchées. Sur la Figure I- 5-, nous proposons une vue d'ensemble des méthodes d'optimisation et de leur schéma global d'optimisation. Le schéma fait apparaitre qu'après une modélisation du problème intervient une étape de modification, où l'on adapte la modélisation du problème à la méthode soit l'inverse ou soit les deux en même temps. On voit aussi qu'on peut utiliser plusieurs méta-heuristiques en même temps ou bien encore, qu'on peut combiner les méthodes approchées avec des méthodes exactes. Ce schéma d'optimisation suppose qu'après la satisfaction d'un critère d'arrêt, toutes les méthodes s'arrêtent en retournant la meilleure solution trouvée (solution optimale, borne supérieure ou borne inférieure). Dans ce qui suit on donne une description des méthodes les plus employées en recherche opérationnelle.

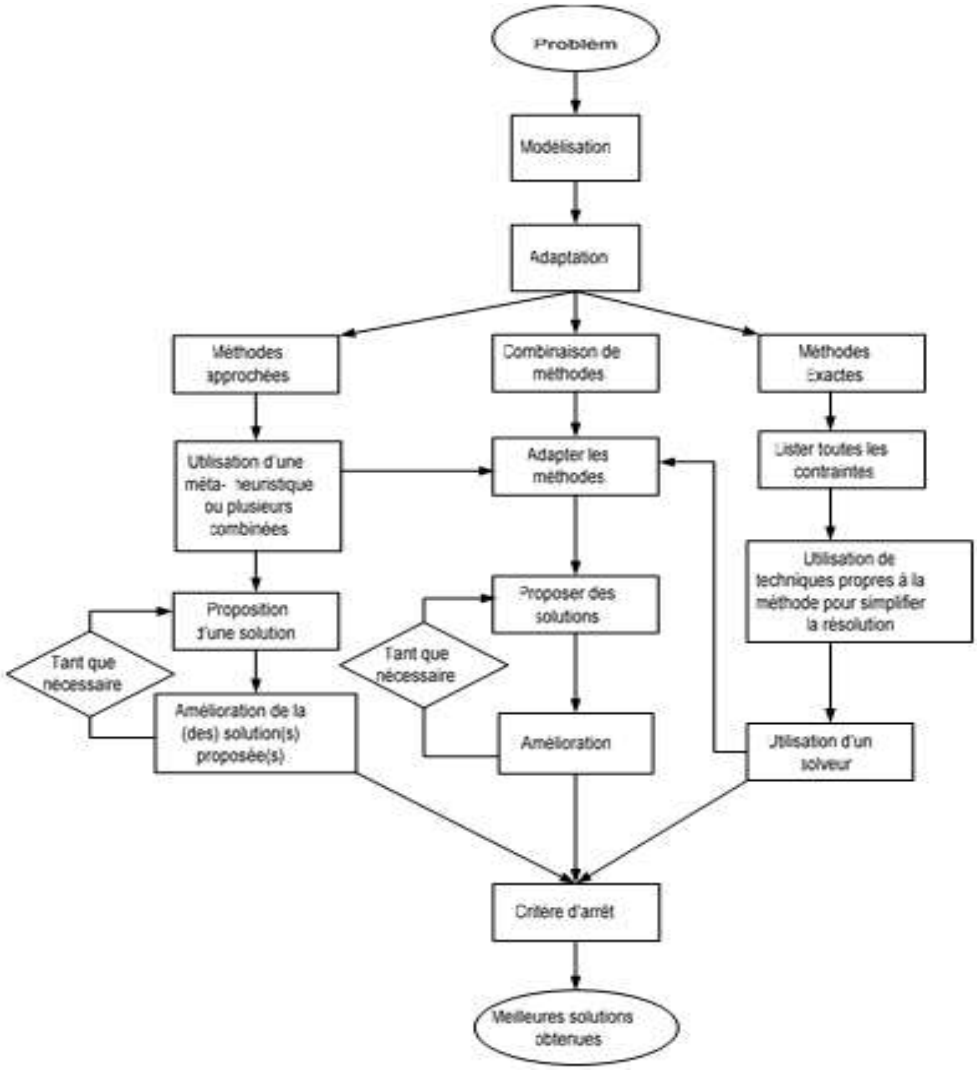


Figure I- 5-schémas d'optimisation

6.1 Les méthodes exactes

Les méthodes exactes sont connues par le fait qu'elles garantissent l'optimalité de la solution mais elles sont très gourmandes en termes de temps de calcul et de l'espace mémoire nécessaire. C'est la raison pour laquelle, elles sont beaucoup plus utilisées pour la résolution de problèmes faciles. La nécessité de disposer d'une solution de bonne qualité (semi optimale) avec un coût de recherche (temps de calcul et espace mémoire) raisonnable a excité les chercheurs à proposer un autre type de méthodes de résolution de problèmes, communément connues par les méthodes approchées.

6.2 Les méthodes approchées

La connaissance de l'espace de recherche des solutions d'un problème permet de guider le Choix d'une méthode. Souvent ces espaces sont énormes et multidimensionnels et le processus de recherche est fastidieux, piégeant ainsi de nombreux algorithmes dans des solutions qui ne sont pas optimales. Ces solutions sont appelées minimums locaux. Le but d'une méthode d'optimisation est donc d'éviter de rester bloquer dans ces endroits (minimums locaux) et d'atteindre des endroits plus prometteurs afin de trouver des solutions optimales, car souvent ils en existent plusieurs. Ces solutions représentent se qu'on appelle communément des minimums globaux. La Figure donne un aperçu de l'espace de rechercheet on remarque sur le graphique l'existence de plusieurs minimums locaux et globaux.

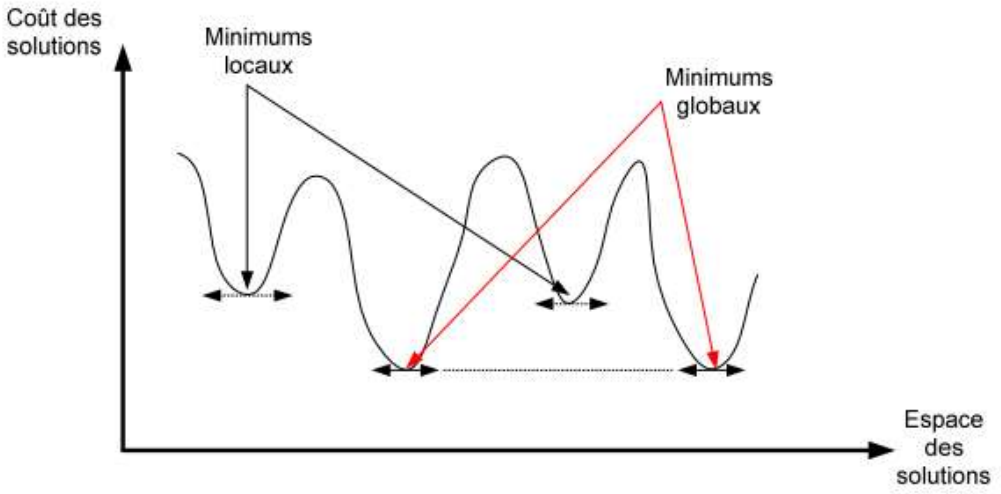


Figure I- 6-Espace de recherche de solutions aux COPs

Ces dernières constituent une alternative aux méthodes exactes. En fait, elles permettent de fournir des solutions de très bonne qualité en un temps de calcul raisonnable. De nombreuses méthodes approchées ont été proposées. Elles sont plus pratiques pour la résolution de problèmes difficiles ou de problèmes dont on cherche des solutions en un bref délai. Ces méthodes sont souvent classées en deux catégories : des méthodes heuristiques et des méthodes méta-heuristiques.

6.2.1 Les heuristiques de construction

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

Définition 1. « Une heuristique (règle heuristique, méthode heuristique) est une règle d'estimation, une stratégie, une astuce, une simplification, ou tout autre type de dispositif qui limite considérablement la recherche de solutions dans des espaces problématiques importants. Les heuristiques ne garantissent pas des solutions optimales. En fait, elles ne garantissent pas une solution du tout. Tout ce qui peut être dit d'une heuristique utile, c'est qu'elle propose des solutions qui sont assez bonnes la plupart du temps. » [63].

Définition 2. « Une méthode heuristique (ou simplement une heuristique) est une méthode qui aide à découvrir la solution d'un problème en faisant des conjectures plausibles mais faillible de ce qui est la meilleure chose à faire. » [63].

Définition 3. « Une heuristique est une règle d'estimation, une stratégie, une méthode ou astuce utilisée pour améliorer l'efficacité d'un système qui tente de découvrir les solutio ns des problèmes complexes. » [64].

Définition 4. « Les heuristiques sont des règles empiriques et des morceaux de connaissances, utiles (mais non garanties) pour effectuer des sélections différentes et des évaluations. [65].

Définition 5. « Les heuristiques sont des ensembles de règles empiriques ou des stratégies qui fonctionnent, en effet, comme des règles d'estimation. » [66].

Définition 6. « Les heuristiques sont des critères, des méthodes ou des principes pour décider qui, parmi plusieurs d'autres plans d'action promet d'être le plus efficace pour atteindre un certain but. » [67].

6.2.2 Les méta-heuristiques

Les méthodes dites métaheuristiques sont des méthodes générales, des heuristiques polyvalentes applicables sur une grande gamme de problèmes. Elles peuvent construire une alternative aux méthodes heuristiques lorsqu'on ne connaît pas l'heuristique spécifique à un problème donné. [68].

Définition « Une métaheuristique est un processus itératif qui subordonne et guide une heuristique, en combinant intelligemment plusieurs concepts pour explorer et exploiter tout l'espace de recherche. Des stratégies d'apprentissage sont utilisées pour structurer l'information afin de trouver efficacement des solutions optimales, ou presque optimales. » [68].

7. Problème d'ordonnancement de job shop

Dans cette partie, nous présentons le problème de job-shop. Nous nous intéressons à ce problème parce qu'il est classique dans la littérature et car il s'agit d'un problème très « général »parmi les problèmes classiques d'ordonnancement d'atelier.

Le problème de job-shop généralise les problèmes classiques d'ordonnancement tels que le problème à une machine, le flow-shop de permutation et le flow-shop général.

7.2 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 **[21]**. Ce problème résista pendant près de 25 ans et engendra une compétition entre les chercheurs du domaine **[22].**Dans cette article les auteurs étudier un cas du problème de job shop et donnée une définition du problème. Ils cherchent sur les méthodes de résolution des problèmes depuis à les équations linéaires aussi la modélisation graphique par un 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 à la construction et l'exécution d'un algorithme exact et les méthodes de branchement et les inégalités valides**; [23].**
- les méthodes approchées pour lesquelles 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. **[23].**

Dans **[24]** on trouve, une bonne explication pour le job shop de son début à la fin des années 90. Il est préparé un liste les articles qui classer par les méthodes de résolution utilisé pour le job shop. En plus la description des méthodes, Jain et Meerran proposent une liste

complète d'instances. Cette dernière sert de base de comparaison entre les méthodes de résolution. Il aussi donnée 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 premières personnes commencées le travail sur les méthodes exactes est **[25]**, puis **[26]** suivi en 1968 par Greenbergqui utilisa une formalisation linéaire en nombre entiers **[27]** S'en suivi alors de nombreuses publications notamment celles de Ficher dans **[28]** et **[29]** qui utilisa les multiplicateurs de Lagrange. Dès les années 1975 McMahon et Florian développent un algorithme **[30]** dépassant la performance des algorithmes de Fisher.

Durant les vingt dernières années furent créés des algorithmes pour résoudre des problèmes plus proches de la réalité avec plus d'une cinquantaine de machines et centaine de jobs. Cet algorithme fut travaillé afin d'associer à chaque contrainte relâchée une pénalité appelée technique de relaxation lagrangienne augmenté. Après, en 1989, **[31]** décrivent un algorithme basé sur le Branch and Bound, qui pour la première fois, donnée une résolution optimale de problème 10x10de Fisher et Thompson aussi que d'autre plus complexes.

7.3 L'ordonnancement de job shop

Sur les ateliers de type job chop les tâches l'exécution des tâches n'est pas la même ordre. En effet chaque tâche passée sur un chemin qui lui est propre.

Ce type d'organisation correspond généralement à une production par lot, notamment dans une unité de production disposant de moyens polyvalent utilisés suivent des séquences différentes afin de réaliser des produit divers **[32].**

Si, pour chaque machine, on dispose d'un et un seul exemplaire, on dit que l'organisation est un Job Shop simple.

Si, au contraire, pour un au moins des machines (postes de travail), on dispose de plus d'un exemplaire, on l'appelle Job Shop Hybride.

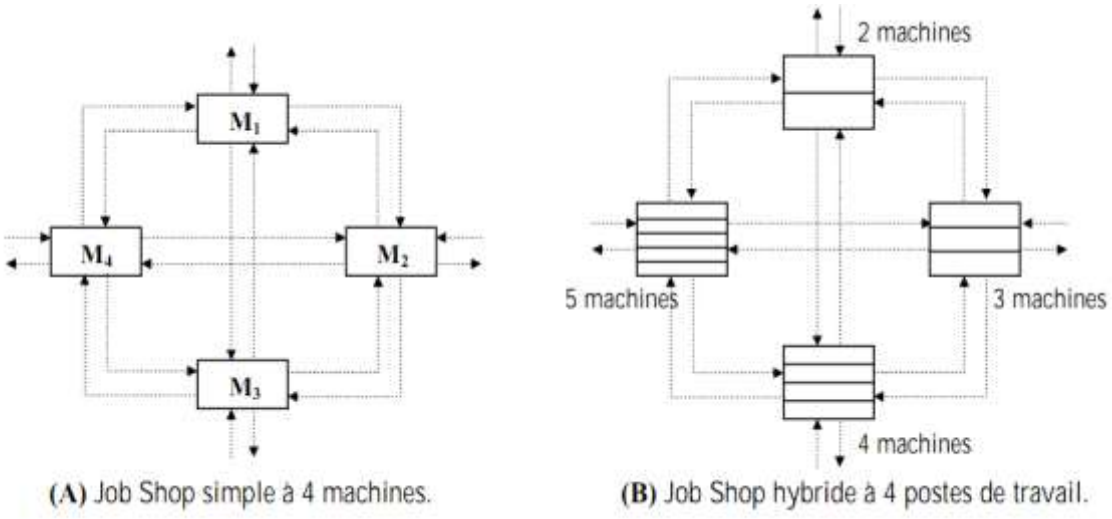


Figure I- 7-atelier job shop simple et hybride

7.4. Définition formelle de problème d'ordonnancement job shop

7.4.2 Job shop classique

Le problème de type job-shop est l'un des problèmes les plus étudiés dans la littérature de l'ordonnancement. Son importance théorique ainsi que la modélisation de nombreuses applications industrielles sous forme de systèmes de type job-shop le rendent très intéressant.

La particularité des problèmes de type job shop par rapport aux problèmes de production de type flow shop est la gamme opératoire qui n'est pas fixe (atelier à cheminement libre), ainsi que le nombre d'opérations qui n'est pas forcément le même pour tous les jobs. Ces systèmes (problèmes de type job-shop) sont considérés comme des systèmes fortement combinatoires. Ils sont composés d'un ensemble de n jobs, chacun composé d'un ensemble de ni opérations qui peuvent être exécutées sur m machines en respectant les contraintes suivantes :

- 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.
- Une machine ne peut exécuter qu'une seule opération à la fois et à l'instant initial toutes les machines sont disponibles.
- Une opération ne peut être exécutée que sur une seule machine et sans interruption.
- La capacité de stockage entre les machines est considérée comme infinie.
- La préemption des opérations n'est pas autorisée.

La figure suivante représente un problème type de job shop classique composé de 3 jobs et 5 machines dont les gammes opératoires sont :

- J1 : M1, M2, M3, M4, M5.
- J2 : M1, M5, M4.
- J3 : M2, M3, M4

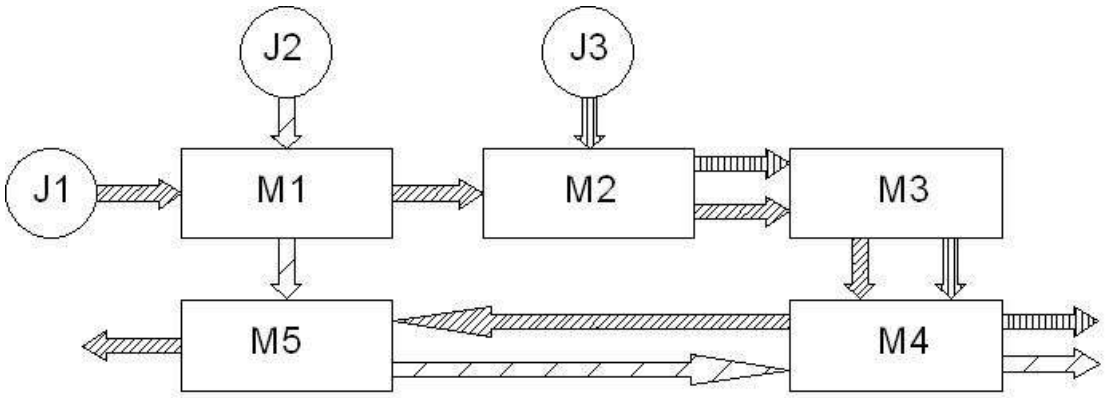


Figure I -8-Problème de job shop classique composé de 3 jobs et 5 machines

7.4.2 Job-Shop flexible

Une façon d'augmenter la productivité d'un atelier est d'augmenter sa flexibilité. Pour cela, nous pouvons multiplier le nombre de machines qui réalisent la même tache. Le modèle résultant est connu dans la littérature comme un job-shop hybride. Dans ce modèle, les machines qui effectuent la même opération sont groupées dans un même étage.

Les problèmes de type job-shop hybride (JSH) sont une extension de deux problèmes d'ordonnancement : le problème de job-shop et le problème d'ordonnancement des machines parallèles. Le JHS peut être vu comme un problème de job-Shop qui possède une ou plusieurs machines parallèles par étage. La machine nécessaire pour exécuter une opération n'est pas connue a priori. Par conséquent, le problème se compose de deux parties dont la première est de trouver la séquence des jobs sur les étages, la deuxième partie est de trouver la machine nécessaire à l'exécution d'un job tout on respectant les contraintes du job-shop.

Un exemple de ce problème à m étages et trois machines maximum par étage, est donné dans la (Figure I -9-).

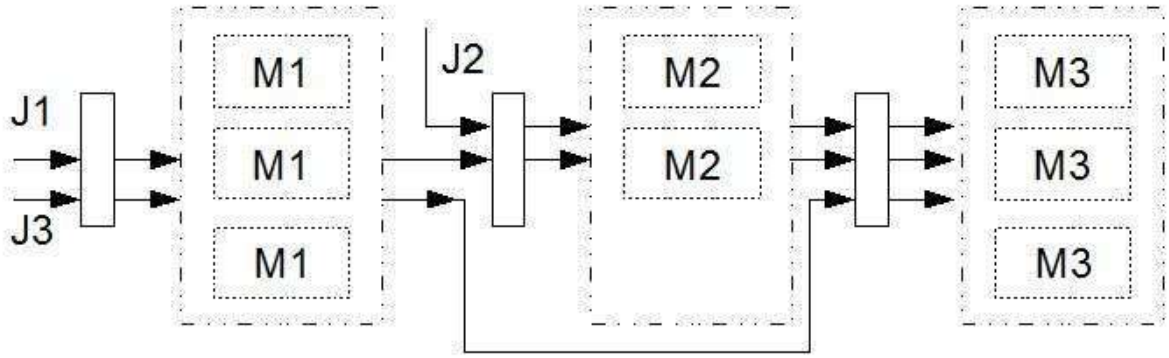


Figure I -9-Représentation d'un système de type Job-shop hybride à trois étages

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.

8. Conclusion :

Nous avons abordé dans ce chapitre l'aspect commun des problèmes d'ordonnancement en production, en exposant notamment les notions de base concernant ces problèmes. D'abord, le cadre général de l'ordonnancement comme fonction originale du système de gestion de production a été évoqué, en présentant succinctement les systèmes de production, la gestion de production et le rôle de l'ordonnancement au sein de ces systèmes.

Le deuxième point exposé dans ce chapitre, est la caractérisation du problème d'ordonnancement d'atelier en présentant sa définition, et en précisant notamment les différents éléments qui le déterminent, à savoir : les tâches, les ressources, les contraintes et les critères d'optimisation ; ainsi que les approches de classification qui s'appuient sur un ensemble de paramètres portant sur différentes caractéristiques du problème dont l'organisation d'ateliers Job Shop, Flow Shop, Open Shop) décrites par la suite, constitue un champ important de la classification.

Nous avons défini aussi les Métaheuristiques car elles représentent une bonne alternative de résolution pour le problème d'ordonnancement du job shop qui est un problème d'optimisation de nature combinatoire et du type NP-difficile.

CHAPITRE 2

LES PROBLEMES D'ORDONNANCEMENT ET LES METHODES DE RESOLUTION

1. INTRODUCTION

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

Dans ce chapitre, nous présenterons les différentes méthodes pour la recherche d'une bonne solution aux problèmes d'ordonnancement. Nous présenterons le principe de chaque méthode d'optimisation appliquée en ordonnancement.

2. Classification des Problèmes d'ordonnancement

Etant donné la diversité des problèmes d'ordonnancement, nous utilisons couramment un formalisme de classification [47],[48], permettant de distinguer les problèmes d'ordonnancement entre eux et de les classer. Ce formalisme, comporte trois champs : γ, β, α, permettant de décrire les différentes entités d'un problème d'ordonnancement.

2.1. Champα : Organisation des ressources

Le champ « α » est composé de deux sous champs α1, α2 . Les ressources peuvent être parallèles (chaque machine peut exécuter chaque tâche) ou dédiées (spécialisées à l'exécution d'une ou plusieurs tâches). Dans le cas de machines parallèles, elles peuvent être soit identiques et avoir la même vitesse pour toutes les tâches, soit uniformes : chaque machine a alors sa propre vitesse qui ne dépend pas de la tâche

Exécutée, soit non liées (cas plus général) où la vitesse d'exécution dépend de la machine et de la tâche exécutée.

Le paramètre $\alpha_1 \in \{\varphi$ J, O, F, R, Q, P$\}$, représente le type de ressources principales (machines) utilisées :

$\alpha_1 = \varphi$: une seule machine. C'est le cas le plus simple qui peut être vu comme un cas particulier des autres configurations.

- α_1 = P: machines parallèles identiques. Les ressources sont composées de machines de même cadence, disposées en parallèle pouvant exécuter toutes les tâches.

- α_1= Q : machines parallèles uniformes. Les cadences des machines sont différentes deux à deux, mais restent indépendantes des tâches.

- α_1= V: machines parallèles non liées. Les cadences des machines sont différentes deux à deux et dépendent des tâches exécutées.

- α_1 = F : fow-shop. Les tâches sont décomposées en plusieurs opérations qui doivent être exécutées sur l'ensemble des machines, disposées en série, selon un même routage.

- α_1= O : open-shop. Les opérations des tâches doivent être exécutées sur certaines machines sans restriction sur le routage des tâches.

- α_1= J : job-shop. Les opérations des tâches doivent être exécutées sur l'ensemble des machines disposées en série, mais peuvent avoir des routages différents.

Le paramètre $\alpha_2 \in \{\varphi, m\}$, dénote le nombre de machines composant le système.

Lorsque $\alpha_1 = \varphi$, le nombre de machines est variable, tandis que lorsque $\alpha_1 = m$ le nombre de machines est égal à m (m> 0).

Le champ β contraintes et caractéristiques du système :

Ce champ $\beta = \beta_1\ \beta_2\ \beta_3\ \beta_4\ \beta_5\ \beta_6\ \beta_7\ \beta_8$ décrit les caractéristiques des ressources et des taches. Pour répondre à des problèmes industriels des plus divers, la variété de ces caractéristiques est très importante.

Tout d'abord, différents modes d'exécution sont possibles : soit avec préemption (l'exécution d'une opération peut être interrompue puis reprise sur la même machine ou sur une autre), soit sans préemption (lorsqu'une opération est commencée, elle doit être terminée avant de pouvoir exécuter une autre opération sur la même machine).

Le champ γ: critères d'optimisation qui concerne les critères d'évaluation d'un ordonnancement. Les objectifs visés sont liés à une bonne utilisation des ressources, une minimisation du délai global ou encore le respect d'un maximum de contraintes. Nous donnons ici les critères les plus couramment utilisés :

- $\gamma = C_{Max}$: Makespan. Ce critère vise à minimiser la date de sortie du système de la dernière tâche ($\gamma = C_{Max} = Max_j\, C_j$, C_j représente la date fin d'exécution de la tâche j)

- $\gamma = w_j C_j$: Somme pondérée des dates de fin d'exécution. Ce critère représente la somme des encours des tâches dans le système.
- $\gamma = L_{Max}$: Décalage temporel maximal. Ce critère mesure la plus grande violation des dates d'échéances souhaitées ($L_{Max} = Max_j\ L_j = Max_j(C_j - d_j)$)
- $\gamma = \sum w_j T_j$: Somme pondérée des retards ($T_j = Max_j(C_j - d_j, 0)$) qui permet d'évaluer les pénalités dues aux retards (w_j : poids de la tache j).
- $\gamma = \sum w_j U_j$: Somme pondérée du nombre de tâches en retard ($U_j = 1$ si la tâche j est en retard, 0 sinon).

3. Complexité :

L'expérience montre que certains problèmes sont plus faciles que d'autres à résoudre. Une théorie de la complexité a été développée et permet mathématiquement de classer les problèmes faciles et difficiles en deux classes : les classes P et NP [47]. Nous exposerons dans la partie qui suit, les grands principes de la théorie de la complexité des problèmes d'ordonnancement.

4. Les classes P et NP :

Pour pouvoir exposer la notion de classe de problèmes, il est tout d'abord nécessaire de distinguer les problèmes de décision des problèmes d'optimisation. Un problème de décision est un problème pour lequel la réponse est "oui" ou "non". Il est possible d'associer à chaque problème d'optimisation, un problème de décision en introduisant un seuil k correspondant à la fonction objectif f. Le problème de décision devient : "existe-t-il un ordonnancement réalisable (S) tel que $f(s) \leq k$?"

Il est alors possible de définir la classe P qui regroupe les problèmes de décision résolubles par des algorithmes polynomiaux [48].

Un algorithme polynomial est défini comme un algorithme dont le temps d'exécution est borné par O (p(x)) ou p est un polynôme et xla longueur d'entrée d'une instance du problème. Les algorithmes dont la complexité ne peut pas être bornée polynomialement sont qualifiés d'exponentiels.

La classe NP regroupe les problèmes qui peuvent être résolus en temps polynomial par un algorithme non déterministe (Algorithme qui comporte des instructions de choix) [49], [48]. Pour ces algorithmes, si à chaque instruction le bon choix est effectué, le temps de calcul est polynomial. Si au contraire tous les choix sont énumérés, l'algorithme devient déterministe et son temps de calcul devient exponentiel.

Les algorithmes "ordinaires" sont évidemment des cas particuliers des algorithmes non déterministes [48]. Aussi tout problème de décision qui peut être résolu par un algorithme polynomial, et qui donc appartient à la classe P, appartient également à la classe NP. D'où P ⊆NP.

Parmi les problèmes de la classe NP, une large classe de problèmes, les problèmes NP-complets, sont équivalents entre eux quant à l'existence d'un algorithme polynomial pour les résoudre. C'est à dire, s'il existe un algorithme polynomial pour résoudre un seul de ces problèmes, alors il en existe un pour chaque problème de la classe NP. Afin de décrire cette classe d'équivalence, définissons tout d'abord la réduction polynomiale entre deux problèmes. Soient P1et P2, deux problèmes de décision. On dit que P1se réduit polynomialement à P2 (noté P1∞P2) s'il existe un algorithme de résolution de P1, qui fait appel à un algorithme de résolution de P2, et qui est polynomial lorsque la résolution deP2est comptabilisée comme une opération élémentaire. On dit alors qu'un problème de décision est NP-complet si tout problème de la classe NP se réduit polynomialement à lui [49]. Les problèmes d'optimisation combinatoire dont le problème de décision associé est NP-complet sont qualifiés de NP-difficiles.

4.1 La preuve de l'appartenance à la classe NP-difficile

Le problème de Job Shop est classé par la théorie de complexité parmi les problèmes NP-difficiles au sens fort. Afin de donner une idée sur l'importance de l'espace de solutions pour une instance du problème, rappelons-nous qu'il existe $\mathbf{(n\ !)^m}$ différentes solutions pour un problème de n tâches à réaliser sur m machines.

- Pour un problème de 4 tâches et 5 machines (4×5) : (n!)m = 7962624
- Pour un problème de 5 tâches et 4 machines (5×4) : (n!)m = 207360000

La difficulté du Job Shop est fonction du nombre de tâches, du nombre de machines, du nombre d'opérations par tâche, et de la durée des opérations.

5. Méthodes de résolution :

Nous avons vu précédemment qu'il existe des problèmes d'ordonnancement de complexité différente. Les problèmes appartenant à la classe Pont des algorithmes polynomiaux permettant de les résoudre. Ces algorithmes sont spécifiques au problème à résoudre et nous ne les citerons pas ici. Pour les problèmes appartenant à la classe NP, l'existence d'algorithmes polynomiaux semble peu réaliste. Ainsi, différentes méthodes de résolution (méthodes exactes ou heuristiques), sont largement utilisées pour appréhender les problèmes NP-difficiles. Nous exposons dans cette partie, les grandes lignes des méthodes les plus connues classées en les méthodes exactes, les méthodes approche .

5.1 **Méthodes exactes :**

Ces méthodes sont basées soit sur une résolution algorithmique ou analytique, soit sur une énumération exhaustive de toutes les solutions possibles. Elles s'appliquent donc aux problèmes qui peuvent être résolus de façon optimale et rapidement. Les méthodes exactes ont l'avantage d'obtenir des solutions dont l'optimalité est garantie. Ces méthodes sont génériques et demandent une particularité vis-à-vis d'un problème spécifique. Parmi les méthodes exactes, on peut citer :

- La programmation dynamique établie par Bellman (1954). Cette méthode a été appliquée pour la résolution de certains problèmes d'optimisation à l'aide de formules de récurrence.

- Le Branch an Bound (Procédure de séparation et évaluation), consistant à appliquer une technique de séparation et évaluation. La séparation permet de décomposer le problème en un ensemble de sous-problèmes de taille réduite. Les sous problèmes résultants sont ensuite évalués d'une relaxation (continue ou lagrangienne principalement) jusqu'à ne plus avoir que des problèmes faciles à résoudre ou dont on sait avec certitude qu'ils ne peuvent pas contenir de solution optimale [50].

Les méthodes exactes ont l'avantage de fournir une solution optimale au problème traité, mais elles sont généralement couteuses en terme du temps de calcul pour les problèmes de types NP-Difficiles. Au contraire, les méthodes approchées visent à générer des solutions de haute qualité en un temps de calcul raisonnable, mais il n'existe aucune garantie de trouver la solution optimale. Ces méthodes approchées sont-elles mêmes divisées en deux sous- catégories : les algorithmes d'approximation et les heuristiques. Contrairement aux heuristique, les algorithmes d'approximation garantissent la qualité de la solution trouvée par rapport à

l'optimal. Enfin, il existe encore deux sous- classes de méthodes heuristiques : les heuristiques spécifiques à un problème donné, et les Métaheuristiques qui regroupent des méthodes plus génériques.

5.2 Méthodes approchées :

Depuis la fin des années 80, les méthodes approchées, et plus particulièrement les métaheuristiques, suscitent un intérêt croissant pour leur capacité à fournir des solutions d'excellente qualité pour une consommation en temps de calcul réduite. Une méthode approchée ou heuristique est un algorithme d'optimisation qui a pour but de trouver une solution réalisable de la fonction objectif, mais sans garantir d'optimalité. Le principal avantage de ces méthodes est qu'elles peuvent s'appliquer à n'importe quelle classe de problèmes, faciles ou très difficiles, bien ou mal formulés, avec ou sans contraintes. En particulier, elles ne nécessitent pas une modélisation mathématique du problème.

5.2.1 Les heuristiques

Elles sont basées sur des méthodes empiriques, elles utilisent des règles simples pour optimiser un ou plusieurs critères. Le principe général de cette catégorie de méthodes est d'intégrer des stratégies de décisions pour construire une solution proche de l'optimal tout en essayant d'avoir un temps de calcul raisonnable [51].

Exemples d'heuristiques :

FIFO : firset in firset out (le premier entre est le premier sortir)

SPT (Shortest Processing Time) : La prochaine opération ordonnancée est celle dont la durée opératoire est inférieure à celles des autres opérations.

LPT (Longest Processing Time) : La prochaine opération ordonnancée est celle dont la durée opératoire est supérieure à celles des autres opérations.

5.2.2 Les métaheuristiques :

Le mot métaheuristique est composé de deux mots ; le mot méta qui est un préfixe signifiant "au-delà" ou bien "dans un niveau supérieur"; et le mot heuristique qui vient du verbe heuriskein et qui signifie 'trouver' [52].

Dans la littérature on trouve des définitions qui expliquent bien la notion de ' métaheuristiques', citons :

"A metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions."[1]

"A metaheuristic is an iterative master process that guides and modifies the operations of subordinate heuristics to efficiently produce high-quality solutions. It may manipulate a complete (or incomplete) single solution or a collection of solutions at each iteration. The subordinate heuristics may be high (or low) level procedures, or a simple local search, or just a constructive method."[2]

"A metaheuristicis a set of concepts that can be used to define heuristic methods that can be applied to a wide set of different problems. In other words, a metaheuristic can be seen as a general algorithmic framework which can be applied to different optimization problems with relatively few modifications to make them adapted to a specific problem. Examples of metaheuristics include simulated annealing (SA), tabu search (TS), iterated local search (ILS), evolutionary algorithms (EC), and ant colony optimization (ACO)."[1]

A partir de ces définitions, on peut dire que les métaheuristiques sont des stratégies qui permettent de guider la recherche d'une solution optimale, leur but est de faire explorer l'espace de recherche d'une manière efficace pour atteindre des solutions presque optimales ;

Elles sont en général non déterministes (pas de garantie), généralement, elles contiennent des techniques pour échapper de l'optimum local, leurs concepts de base peuvent être décrits de manière abstraite (indépendamment d'un problème spécifique). Même si on utilise des heuristiques qui tiennent compte de la spécificité du problème traité, ces heuristiques resteront contrôlées par une stratégie de niveau supérieur [52].

5.2.2.1 Classification

On peut classer les métaheuristiques selon plusieurs points de vue ; par conséquent on peut trouver une métaheuristique qui appartient aux différentes classes selon le critère de la classification [53], donc on a comme critères :

- **Le nombre des solutions manipulées à la fois**: selon ce point de vue, on distingue deux classes des métaheuristiques :

a. Les métaheuristiques basées sur la manipulation (évaluation et modification) d'une solution à la fois. Dans ce type de méthodes la notion de voisinage d'une solution est pertinente. On

trouve dans cette classe la méthode de recuit simulé, la recherche tabou, la recherche locale itérée, la recherche avec voisinage variable et autres.

b. Les métaheuristiques basées sur la manipulation d'un ensemble de solutions à la fois dite généralement population de solutions; citons dans cette classe les algorithmes génétiques, l'optimisation par colonies de fourmis et l'optimisation par essaim des particules.

• **Type de parcours**: les métaheuristiques sont des méthodes qui explorent l'espace de recherche soit avec le déplacement dans l'ensemble de voisinage ou avec l'échange de l'état des individus d'une population. Ce déplacement forme une sorte de parcours qui peut être continu comme dans le recuit simulé et la recherche tabou ; ou discontinu comme dans la recherche locale itérée, les algorithmes génétiques, l'optimisation par colonies de fourmis et GRASP.

• **Emploi de mémoire**: l'emploi de la mémoire signifie l'utilisation de l'historique de la recherche pour déduire l'état courant de l'optimisation. Dans ce contexte, on peut distinguer deux types : les métaheuristiques dites ''sans mémoire'' où l'état actuel de la recherche est déterminé uniquement suivant l'état précédent, c'est le cas de la descente récursive, le recuit simulé. Par contre, il existe des méthodes qui utilisent l'historique pour guider la recherche telle que la recherche tabou. la mémoire peut être figurée selon différentes formes, par exemple, on peut considérer que l'ensemble des individus d'un algorithme génétique forme une mémoire, la phéromone dans l'optimisation par colonies de fourmis aussi et les positions dans l'optimisation par essaim des particules.

• **La Source de la méthode**: on trouve généralement deux classes : d'un côté les méthodes inspirées de la nature tel que le recuit simulé, les algorithmes génétiques, l'optimisation par colonies de fourmis et l'optimisation par essaim des particules, et de l'autre côté, on trouve des méthodes qui ont des origines non naturelles, telle que la recherche avec tabou, la recherche avec voisinage variable, la recherche locale itérée et GRASP.

• **Nombre et structure de voisinage** : le voisinage est l'ensemble des solutions (états) atteignables à partir de la solution (état) courante ; le voisinage peut être fixe ou variable.

Les métaheuristiques, selon ce critère, sont classifiées en deux : les métaheuristiques avec un voisinage fixe, c'est le cas de la majorité des heuristiques ; et les métaheuristiques avec un voisinage variable, c'est le cas de la recherche avec voisinage variable.

5.3 Les méthodes basées sur une seule solution

Appelées aussi méthodes de trajectoire, le processus de recherche trace une sorte de chemin ou un parcours le long de leur progression, il se déplace d'une solution vers une autre dans l'espace de recherche ; ce déplacement est possible grâce à la notion de voisinage. Les sections suivantes donnent une vision sur les méthodes de recherche locale les plus connues, de la descente récursive qui représente une méthode de base au recuit simulé et la recherche tabou.

5.3.1 Le recuit simulé :

La méthode du recuit simulé a été introduite en 1983 par Kirkpatrick et al., (1983). Cette méthode originale est basée sur les travaux bien antérieurs de Metropolis et al., (1953). Cette méthode que l'on pourrait considérer comme la première métaheuristique 'grandpublic' a reçu l'attention de nombreux travaux et principalement de nombreuses applications [45].

L'analogie historique s'inspire du recuit des métaux en métallurgie : un métal refroidi trop vite présente de nombreux défauts microscopiques, c'est l'équivalent d'un optimum local pour un problème d'optimisation combinatoire. Si on le refroidit lentement, les atomes se réarrangent, les défauts disparaissent, et le métal a alors une structure très ordonnée, équivalente à un optimum global [45].

La méthode du recuit simulé, appliquée aux problèmes d'optimisation, considère une solution initiale et recherche dans son voisinage une autre solution de façon aléatoire. L'originalité de cette méthode est qu'il est possible de se diriger vers une solution voisine de moins bonne qualité avec une probabilité non nulle. Ceci permet d'échapper aux minima locaux [45].

L'évolution de l'algorithme est gérée par un paramètre appelé énergie, l'énergie désigne la valeur de la fonction objectif, dite aussi fonction coût de la solution courante. La température Test un paramètre de l'algorithme : elle détermine la probabilité avec laquelle une dégradation de la solution courante est acceptée. La température décroit graduellement au cours du temps.

Au début de la recherche, la température élevée permet au recuit d'explorer l'espace de recherche sans être piégé par les optima locaux. A mesure que la température décroît, le recuit est de moins au moins capable d'explorer l'espace de recherche, et la recherche se concentre sur une zone déterminée : l'exploration prend le dessus sur l'exploration. Par conséquent la vitesse de décroissance de la température dé termine la vitesse à laquelle le recuit « converge » vers une solution.

La méthode recuite simulée donne de très bons résultats pour le problème de voyageur commerce mais pas pour le problème d'affectation ; leur performance liée au schéma de refroidissement [45].

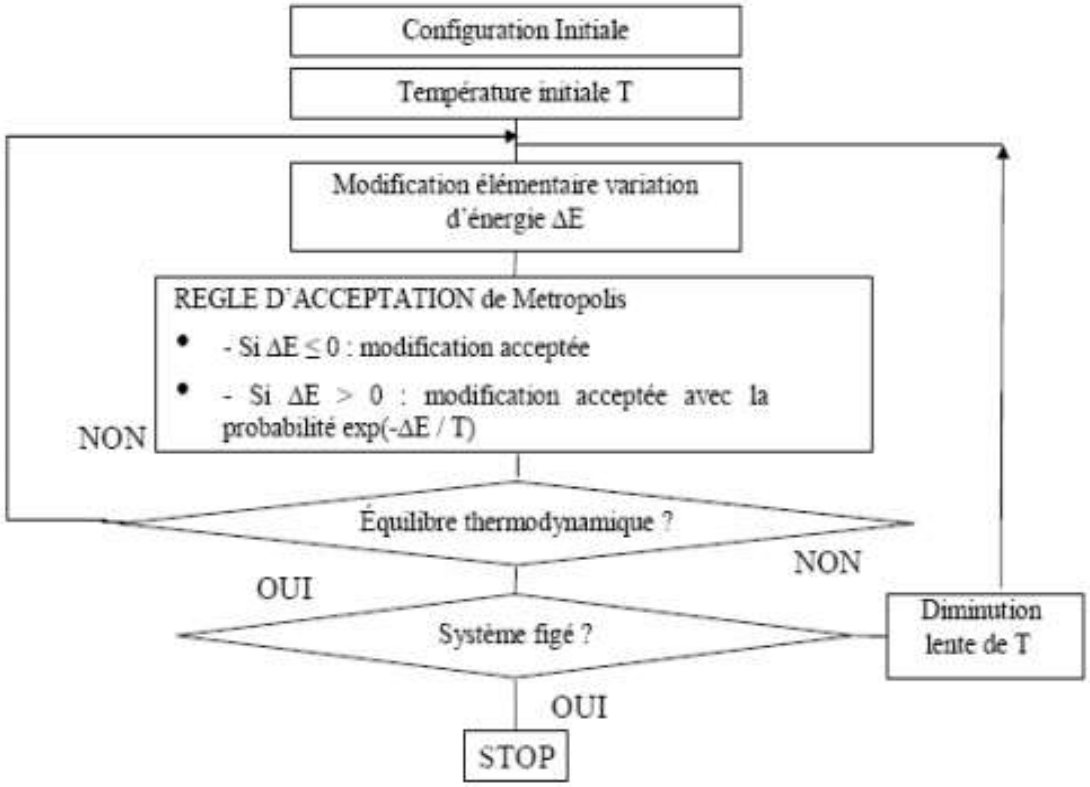


Figure Π- 1-L'organigramme du recuit simulé

5.3.1.1 Les avantages de recuit simulé:

Parmi les avantages de la méthode, on cite:

- Traite les fonctions de coût avec les degrés tout à fait arbitraires de non linéarité, la discontinuité, et l'imprévisibilité.
- Processus assez arbitraire sur les conditions limites et les contraintes imposées sur les fonctions de coût.
- Simple à implémenter
- Garantie statistique de trouver une solution optimale

5.3.1.2 Les inconvénients de recuit simulé

La méthode comprend quelques inconvénients parmi lesquels :

- le non déterminisme

- Difficulté de choisir les paramètres efficaces ; par exemple le schéma de refroidissement.
- Le compromis entre la vitesse et l'optimisation.

5.3.2 La Recherche Tabou

La recherche Tabou est une métaheuristique d'optimisation présentée par Fred Glover en 1986 [54]. L'algorithme a été intitulé recherche tabou du faite que le choix des solutions récemment visitées est pratiquement interdit.

Depuis cette date, la méthode est devenue très populaire, grâce aux succès qu'elle a remportés pour résoudre de nombreux problèmes complexes de très grande taille (souvent NP-durs). Cet algorithme introduit une notion de mémoire dans la stratégie d'exploration de l'espace de recherche (poursuivre la recherche de solutions même lorsqu'un optimum local est rencontré).

La recherche Tabou utilise des opérations itératives locales ou par voisinage pour passer d'une solution à une autre, jusqu'à ce qu'une condition d'arrêt soit satisfaite.

5.3.2.1 Principe de la recherche tabou

On se déplace de solution en solution sur l'espace des solutions en s'interdisant de revenir dans une solution déjà visitée [46], tout en cherchant constamment à améliorer la meilleure solution courante et en conservant en mémoire la liste des précédents déplacements et se servir de cette information afin d'en orienter le déroulement futur.

Le voisinage d'une solution est défini par une transformation élémentaire (mouvement) permettant de passer d'une solution à une autre solution proche avec une petite modification de la structure de la solution [46].

Le processus de la recherche tabou est basé sur :

- L'utilisation de mémoires flexibles (court, moyen, long terme) permettant de bien explorer l'historique de la recherche et le critère d'évaluation.

- La stratégie d'intensification utilisant la mémoire à moyen terme pour encourager la recherche de la solution dans les régions de solutions précédemment parcourues.
- La stratégie de diversification utilisant la mémoire à long terme pour encourager la recherche de la solution dans les régions qui ne sont pas explorées.
- Un mécanisme d'aspiration permet à une solution bien de son statut tabou de redevenir Candidate
-

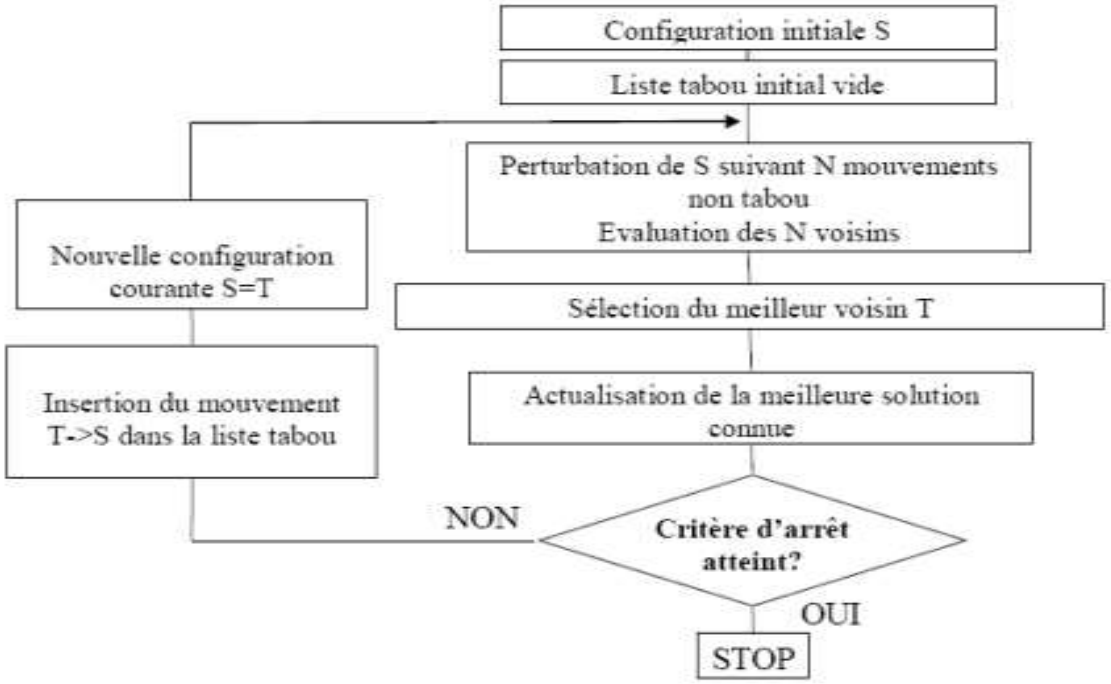


Figure II- 2-L'organigramme de la recherche tabou

5.3.2.2 Applications de la recherche tabou :

On a dit que la méthode de recherche tabou est l'une des métaheuristiques les plus utilisées, elle a été appliquée dans plusieurs domaines : dans le domaine de la télécommunication (conception des réseaux…), la conception (conception assistée par ordinateur, conception des réseaux de transport,…..), le routage (routage des véhicules, voyageur de commerce…), optimisation des graphes (coloration des graphes..), les problèmes d'allocation (problème d'assignement quadratique…) et autres [55].

5.4 Les méthodes basées sur une population des solutions :

Les métaheuristiques basées sur la manipulation d'une population des solutions essayent d'augmenter la qualité moyenne d'un ensemble des solutions via un mécanisme défini. Généralement ces méthodes sont inspirées de la nature, du domaine de la biologie, l'éthologie et autres.

5.4.1 L'optimisation par colonies de fourmis :

L'optimisation par colonies de fourmis ou Ant Colony Optimization (ACO) est une méthode d'optimisation basée sur la manipulation d'un ensemble de solutions ; cette méthode représente toute une classe des métaheuristiques qui reposent sur la notion de l'intelligence collective. La solution finale est plus complexe que celle d'un composant simple. Plusieurs mots apparaissent dans ce domaine : l'auto-organisation, émergence et autres.

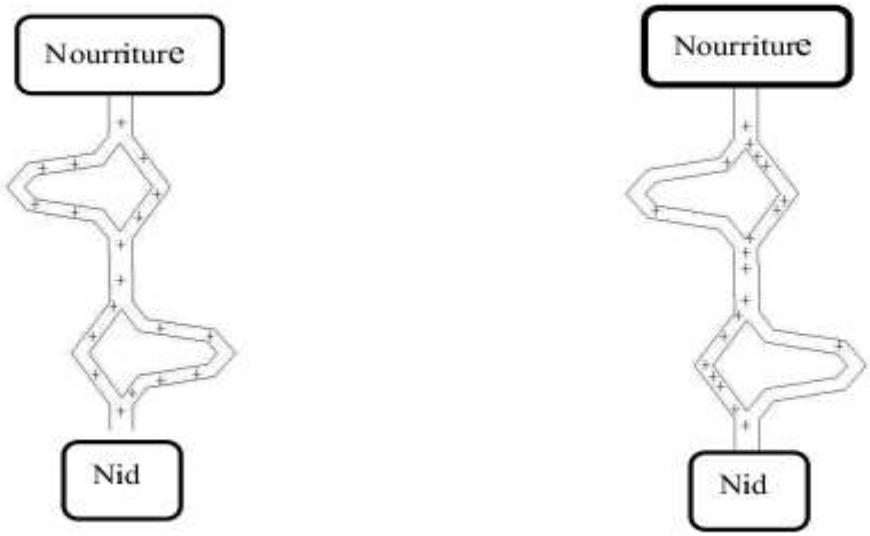


Figure Π- 3-l'influence de l'expérience sur le choix des fourmis

5.4.2 Description et algorithme :

Pour bien comprendre comment fonctionne l'optimisation par colonie de fourmis, on va retourner vers le premier algorithme proposé, c'est l'algorithme « Ant System » proposé par Colorni et autres en 1992 ; au début chaque fourmi est mise aléatoirement sur une ville et elle a une mémoire qui stocke la solution partielle qu'elle a construit jusqu'ici (au commencement la mémoire contient seulement la ville du début).

À partir de sa ville de début, une fourmi se déplace itérativement d'une ville vers une autre mais avec une règle de probabilité.

```
Pour t = 1,…..tmax
   Pour chaque fourmi k=1,….m
        Choisir une ville au hasard
          Pour chaque ville non visitée i
            Choisir une ville j dans la liste jk i  dans les villes restantes, selon la formule (1)*
          Fin Pour
        Déposer une piste Δ (t) sur le trajet conformément à l'équation (2)**
     Fin Pour
     Evaporer les pistes selon la formule  (3)***
Fin pour
```

Figure Π -4-Algorithme de colonies de fourmis de base : the « Ant System »

6. Algorithmes Génétiques :

Cette classe de méthodes est basée sur une imitation des phénomènes d'adaptation des êtres vivants. L'application de ces méthodes aux problèmes d'optimisation a été formalisée par Goldberg en 1989 [56].

Les algorithmes génétiques fonctionnent sur une analogie avec la reproduction des êtres vivants. On part d'une population (ensemble de solutions) initiale sur laquelle des opérations de reproduction, de croisement ou de mutation vont être réalisées dans l'objectif d'exploiter au mieux les caractéristiques et propriétés de cette population. Ces opérations doivent mener à une amélioration (en terme de qualité des solutions) de l'ensemble de la population puisque les bonnes solutions sont encouragées à échanger par croisement leurs caractéristiques et à engendrer des solutions encore meilleures.

* la formule (1) est définit dans l'annexe

** la formule (2) est définit dans l'annexe

*** la formule (3) est définit dans l'annexe

Les difficultés dans l'application des algorithmes génétiques résident dans le besoin de coder les solutions, et dans les nombreux paramètres à fixer (taille de la population, coefficients de reproduction, probabilité de mutation, ...).

6.1 Principe des Algorithmes Génétiques

Dans le cadre des problèmes d'optimisation combinatoires, un individu dans une population représente une solution parmi l'ensemble de solutions possibles pour un problème particulier.

L'application d'un algorithme génétique à un problème particulier nécessite cinq éléments suivants [57] :

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

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

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

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

- **Une fonction à optimiser**

Cette fonction sert à évaluer chaque individu, elle retourne une valeur réelle nommée : « fitness », qui sera utilisée dans le calcul de probabilité de sélection d'un individu [58].

- **Les opérateurs génétiques**

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

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

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

- **Algorithme général**

Les étapes des algorithmes génétiques peuvent varier d'un problème à un autre. Mais, ils se basent sur le même principe. La figure suivante (Figure 2.2), illustre le principe de base des algorithmes génétiques :

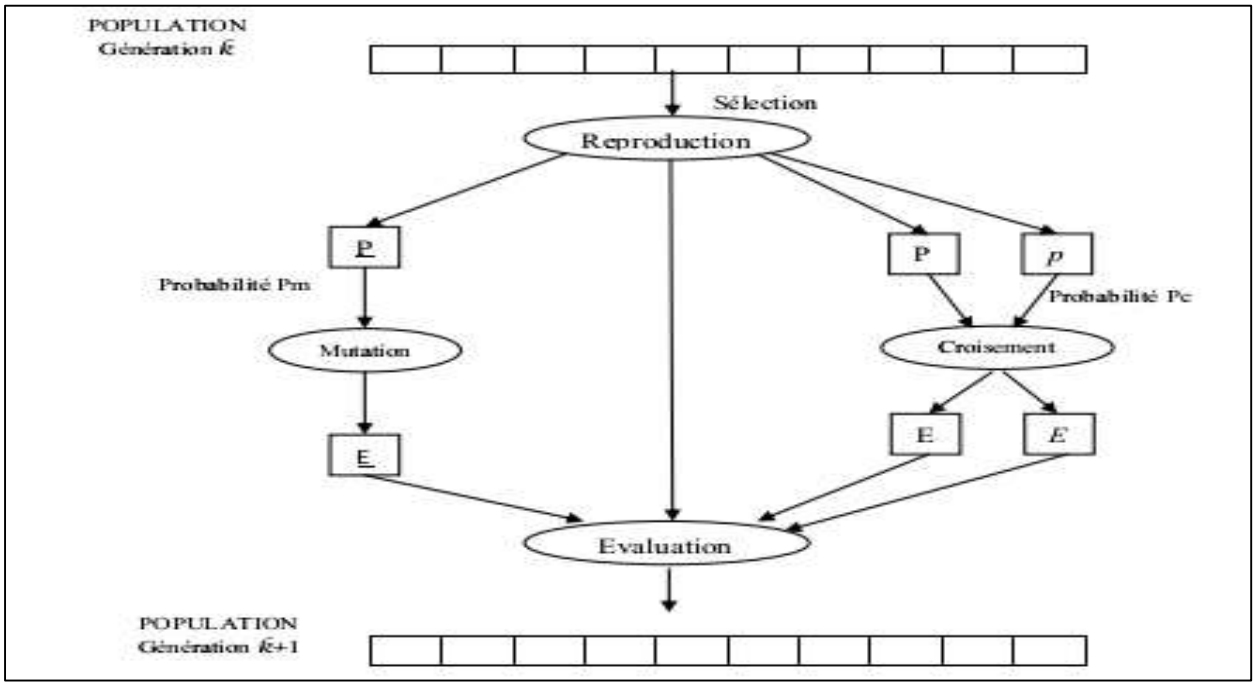


Figure Π -5-Principe général des algorithmes génétiques

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

Début

Génération d'une population initiale de *k* individus ;

Répéter

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

Jusqu'à la satisfaction de la condition d'arrêt.

Retourner la meilleure solution.

Fin

Figure Π -6-Algorithme générale de l'algorithmes Génétiques

6.2 Codage

Les algorithmes génétiques sont appliqués sur une population d'individus, chacun de ces derniers est codé par un *chromosome* ou génotype. Donc, une population est présentée par un ensemble de chromosomes. Le processus de codage d'individus consiste à trouver une structure de données pour les individus.

On distingue plusieurs types de codage, tels que : le codage binaire, le codage par permutation de valeurs entières et le codage par valeur, etc. [57] [59].

6.3 Les opérateurs génétiques

On distingue trois opérateurs génétiques : l'opérateur de sélection, l'opérateur de croisement et celui de mutation.

6.3.1Le croisement

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

Il existe plusieurs stratégies de croisement [57] :

- Le croisement à un point simple
- Le croisement Multipoints
- Le croisement uniforme

6.3.1 La Mutation

L'opérateur de mutation, assure le brassage et la recombinaison des gènes parentaux, permettant de diversifier la future population afin d'éviter une convergence rapide vers une solution optimale localement. Étant aléatoire, cet opérateur agit selon une probabilité Pc fixée par l'utilisateur en fonction du problème à optimiser [60].

On trouve plusieurs stratégies de mutation [57] :

- ✓ **La mutation uni-point**

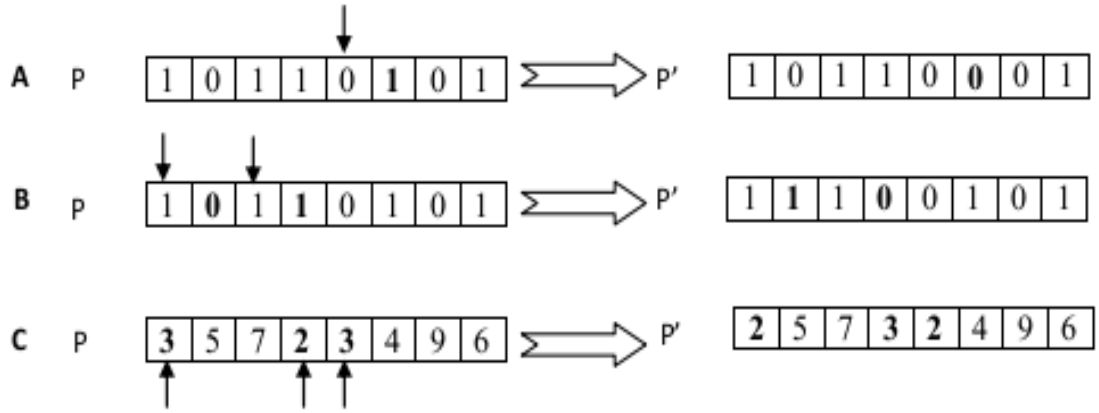
Ce type de mutation se fait par altération d'une seule valeur sur le chromosome.

- ✓ **La mutation bipoints et multipoints**

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

- ✓ **La mutation par valeurs**

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



A : Mutation uni-point. **B :** Mutation bipoints. **C :** Mutation par valeurs 3 et 2.

Figure II -7-Exemple d'opérateurs de mutation.

6.3.2 La sélection

L'opérateur de sélection a pour objectif de choisir les individus qui vont pouvoir survivre et/ou se reproduire pour transmettre leurs caractéristiques à la génération suivante. La sélection se base sur le principe de conservation des individus les mieux adaptés et d'élimination des moins adaptés [61].

6.4 Paramètres de dimensionnement

Il existe plusieurs paramètres prédéfinis qui jouent un rôle crucial dans la performance de chaque algorithme génétique. Parmi ces paramètres de base (on peut trouver d'autres paramètres qui dépendent du problème traité) [62] :

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

- Le nombre de générations peut être défini comme une condition d'arrêt.

6.5 La méthode de résolution de notre problème

Nous avons choisi la méthode algorithme génétique pour résoudre notre problème de job shop presque cette méthode il possédé plusieurs avantages et minime des inconvénients.

6.5.1 Les avantages des algorithmes génétiques :

Les algorithmes génétiques ont plusieurs avantages, citons:

- Ils sont adaptables à plusieurs types des problèmes
- Robustes
- Faciles à implémenter
- Faciles à hybrider
- Faciles à paralléliser

6.5.2 Les inconvénients des algorithmes génétiques :

Parmi les inconvénients des algorithmes génétiques, on note :

- Pas de garantie de convergence
- Temps de calcul important (en cas où la taille de population est grande)

7. Conclusion :

Nous avons présenté, dans ce chapitre, les différentes classes d'un problème d'ordonnancement ainsi que les méthodes de résolution utilisé dans la littérature. Ces méthodes sont regroupées en deux classes principales : les méthodes exactes et les méthodes approchées

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

CHAPITRE 3

RESOLUTION DE PROBLEME D'ORDONNANCEMENT DE JOB SHOP AVEC CONTRAINTE DE TRANSPORT

Introduction

La résolution de problèmes d'optimisation consiste, en règle générale, à définir une ou plusieurs fonctions objectives à minimiser ou maximiser (comme par ex le Makespan) selon le type de problème traité.

Les méta-heuristiques permettent de trouver une ou plusieurs solutions proches de l'optimum pour des problèmes d'optimisations, en tenant compte des fonctions objectifs définies.

Dans ce chapitre nous nous intéressons à la résolution du problème d'ordonnancement de type job shop avec contrainte de transport dont nous allons présenter notre problématique et la démarche de résolution proposée qui est basée sur l'utilisant des méta-heuristiques comme méthode de résolution approchée.

Ce chapitre est composé de trois parties. La première est consacrée à la définition de la problématique. La deuxième présente la démarche suivi et l'algorithme proposé pour résoudre notre problème de job shop. La troisième partie présente les résultats que nous avons obtenus après la simulation sur plusieurs exemples.

2. Formulation du problème

2.1 Description du System à étudier

Le système étudié dans ce mémoire est un système de production de type job shop son principe de fonctionnement est inspiré d'un system réel qui existe dans le laboratoire de productique MELT (université de Tlemcen)

Ce système est constitué de quatre machines (M1, M2, M3 et M4) positionnées sous forme d'un carré Figure III.1

Les produits (pièces) sont stockés dans un stock S1 que l'on suppose d'une capacité illimité

Ces produits sont transportés, devant les machines, dans un sens unidirectionnel à l'aide d'un tapie roulant automatique. Les temps de transport des produits entre machines sont indiqués dans le tableau III.1

Un bras manipulateur est installé à coté de chaque machine qui sert à déplacer les pièces du tapie vers les machines et l'inverse.

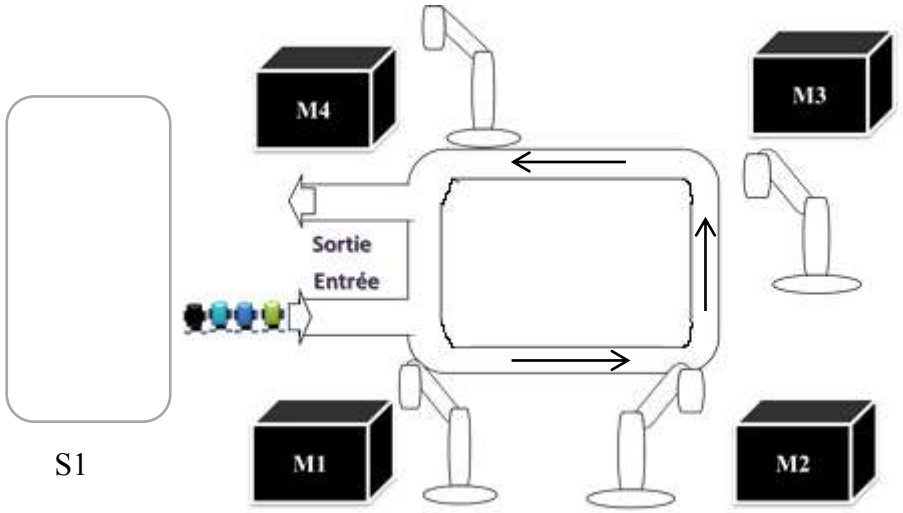


Figure III- 1-le system à étudier (system avec 4 machine

Le tableau suivant regroupe l'ensemble des temps de transport des produits entre les différents postes de travail

Les stations		Temps (seconde)
Départ	Arrivée	
E(entrée)	M1	1.5
M1	M2	5
M2	M3	3
M3	M4	5
M4	M1	3
M4	S(sortie)	1.5

Tableau-III- 1-les temps de transport entre les machines

2.2 Présentation de problème à étudier

Dans ce travail nous nous somme intéressé à un problème d'ordonnancement de type job shop a quatre machines avec une contrainte de transport où les produits sont transportés dans un sens unidirectionnel ce qui diffère aux plusieurs travaux qui ont traité ce type de problème sans tenir compte de cette contrainte.

L'objectif de notre travail est de trouver, en utilisant des méthodes d'optimisation, des meilleures solutions pour un ordonnancement des produits toute en minimisant le temps total d'exécution (Makespan) c'est-à-dire de trouver le meilleur ordonnancement faisable qui minimise le Makespan et qui respect les contraintes imposés.

Le problème est définit donc par **J4/ β1/Cmax** selon la notation la plus utilisée en ordonnancement

- **Fonction objectif** : Minimisation de Makespan (Cmax)

2.3 Les contraintes **:**

1. Une machine ne peut traiter qu'un seul produit à la fois
2. Un produit ne peut s'exécuter que sur une seule machine à la fois
3. Pas de recirculation (un produit ne peut exécuter sur une machine qu'une seule fois)
4. β1 = ∅ : Pas de préemption (le produit doit terminer toute l'opération avant de quitter la machine)
5. Les produits passent dans seul direction
6. Chaque machine possède un fil d'attend avec une capacité illimite.
7. Les produits sont disponibles à t=0.
8. le chariot ne transporte qu'un seul produit à la fois.
9. L'entrée et sortie des produits est la même.

2.4 Hypothèses proposées :

Pour encadrer notre problématique nous avons proposé un ensemble d'hypothèses concernant le fonctionnement de notre système :

- Les produits sont toujours disponibles dans le temps T=0
- Chaque machine possède une file d'attend avec une capacité illimitée
- Quand une machine est entrain de traiter une pièce (P1 par exemple), deux cas sont possibles pour la pièce qui vient juste après dans l'ordre de passage (P2 par exemple):
  1. La pièce (P2) ne demande pas la machine, dans ce cas la pièce peut passer devant la machine sans aucun blocage pour les autres pièces
  2. La pièce (P2) demande la machine dans ce cas la pièce doit attendre le traitement de P1 dans la file d'attente à côté de la machine en libérant donc le passage pour les autres produits.

2.5 Notation

- **k** : la $k^{ème}$ Machine
- **j** : Type de jobs
- $\mathbf{L_j}$: Nombre d'opération par job
- $\mathbf{p_j^k}$: Temps d'exécution de produit (j) dans la machine **k**
- $\mathbf{o_j^k}$: L'opération du job **j** qui sera exécutée dans la machine **k**
- $\mathbf{t_j^k}$: Temps de transport de job **j** vers la machine **k**
- $\mathbf{E_j}$: Temps de fin d'exécution de job
- $\mathbf{D_j}$: Temps de disponibilité du job **j** pour le traiter sur la machine suivante
- $\mathbf{C_k}$: Temps de disponibilité de la machine **k**

3. Démarche de résolution

Dans cette partie nous allons présenter la démarche de résolution ainsi que l'algorithme proposé pour résoudre le problème prés défini, en essayant d'expliquer en détail comment nous avons adopté la méthode Algorithmes Génétiques à notre problème, c'est à dire nous allons présenter le choix du codage, le type de croisement, etc …

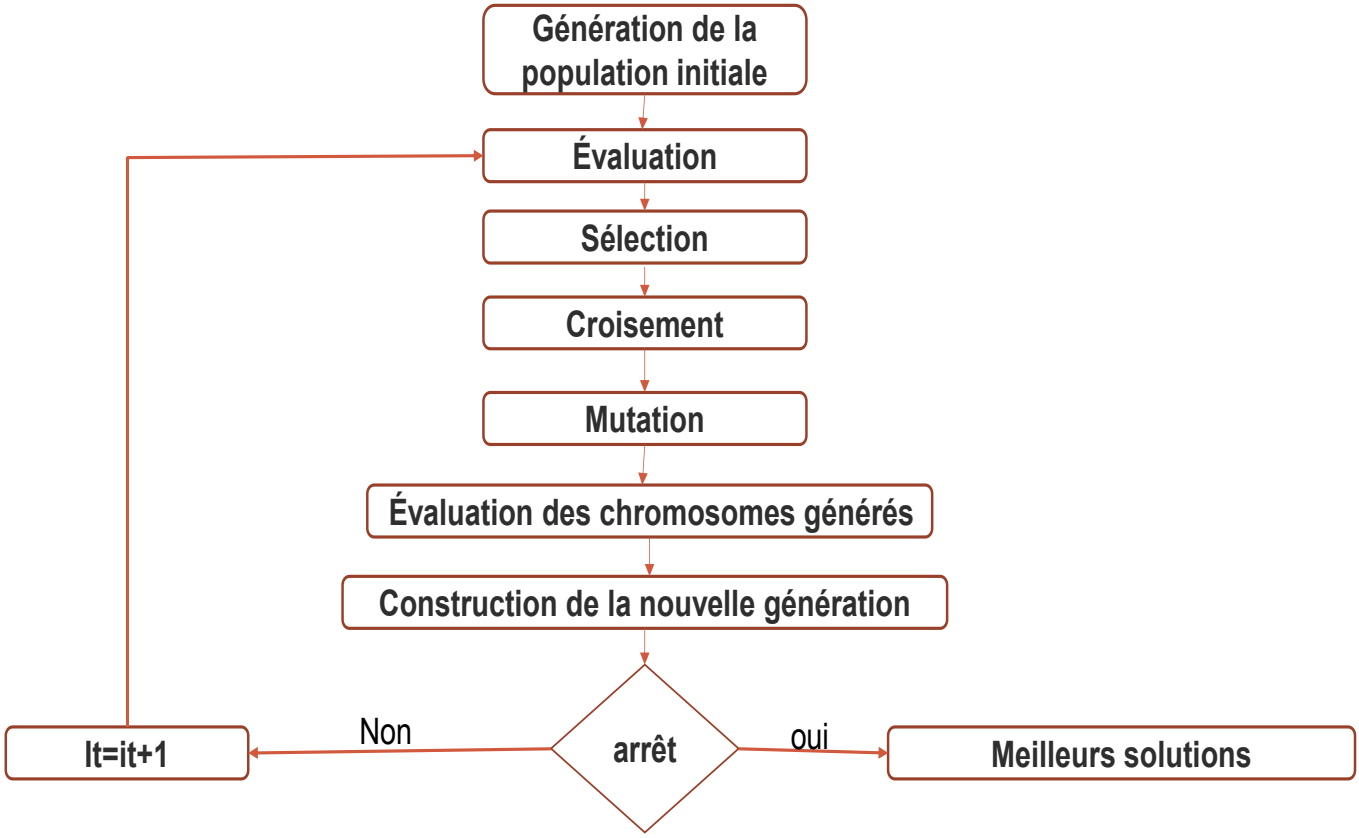


Figure III- 2-Organigramme général de l'Algorithme Génétique implémenté

L'organigramme présenté dans la figure III.2 montre les déférentes étapes de la démarche de résolution utilisée dans notre travail.

3.1 Le codage

Le codage est le déterminant important de l'efficacité de la méthode. Il signifie la transcription d'un ordonnancement admissible en représentation adéquate permettant la réalisation de différents opérateurs génétiques.

Il existe plusieurs types de codage pour notre cas utilisons le codage par job.

3.1.1 Le codage basé sur les jobs :

Ce codage a été utilisé par Holsapple & al [52], il représente une liste ordonnée des Jobs dont la longueur du chromosome est équivalente à *n* gènes ou *n* Jobs. Dans cette représentation, chaque gène représente un Job et l'ordonnancement de tous les Jobs se traduit par le lancement des opérations appartenant au premier Job (premier gène) selon l'ordre décrit

dans la gamme opératoire du premier Job, après ça, les opérations du deuxième Job seront lancées et ainsi de suite jusqu'au dernier Job. Donc, à chaque fois qu'un Job est prêt, toutes ses opérations seront ordonnancées suivant leur ordre dans la gamme opératoire.

Figure III- 3-codage de chromosome

Le chromosome proposé dans la figure III.3 définit d'une part le séquencement de l'entrée des produits dans le système et d'autre part la priorité de traitement du job dans le système c'est-à-dire l'ordre d'exécution des jobs dans les machines comme il est montré dans la figure III.3 (par exemple) le traitement des quatre produits s'effectuer selon l'ordre suivant :

1 ⟶2 ⟶3⟶ 4

Notre choix de ce type de codage est basé sur les points suivants :

- Un seul point d'entrée et un seul point de sortie pour tous les produits
- Le transport des produits s'effectué dans seule direction
- Ce type de codage est facile à appliquer pour notre problème.

3.2 Création de Population initiale

Avant de lancer l'Algorithme Génétique, une population initiale doit être générée. Cet ensemble d'individus qui servira de base pour les générations ultérieures doit être non homogène. Afin de satisfaire cette non homogénéité, la génération de la population initiale dans notre application se fait en générant des chromosomes aléatoires au lieu d'ordonnancements réels.

Algorithme de population initial :

Tant que ch <= taille de population **faire**
Créer nouvelle chromosome chi
Si chi=ch
Créer un chromosome (ch_i) différent à ch.
Fin si
Fin tant que

Algorithme- III- 1Génération de la population initiale

3.3 Croisement

Cet opérateur est très important dans les algorithmes génétiques, mais il est plus intéressant d'utiliser des croisements spécifiques au domaine étudié que d'employer des croisements simples tels que croisement uni-point ou multipoint.
on a choisir le croisement de l'ordre de Job (Jox) .

3.3.1 Le croisement de l'ordre des Jobs (JOX)

Ce croisement consiste à choisir des Jobs aléatoirement, les deux fils héritent les gènes correspondant à ces Jobs en même position dans les parents. Les gènes qui restent de chaque fils sont transférés de l'autre parent, en insérant les gènes ne correspondant pas aux Jobs pré choisis dans leur ordre, aux emplacements vides successifs du fils.

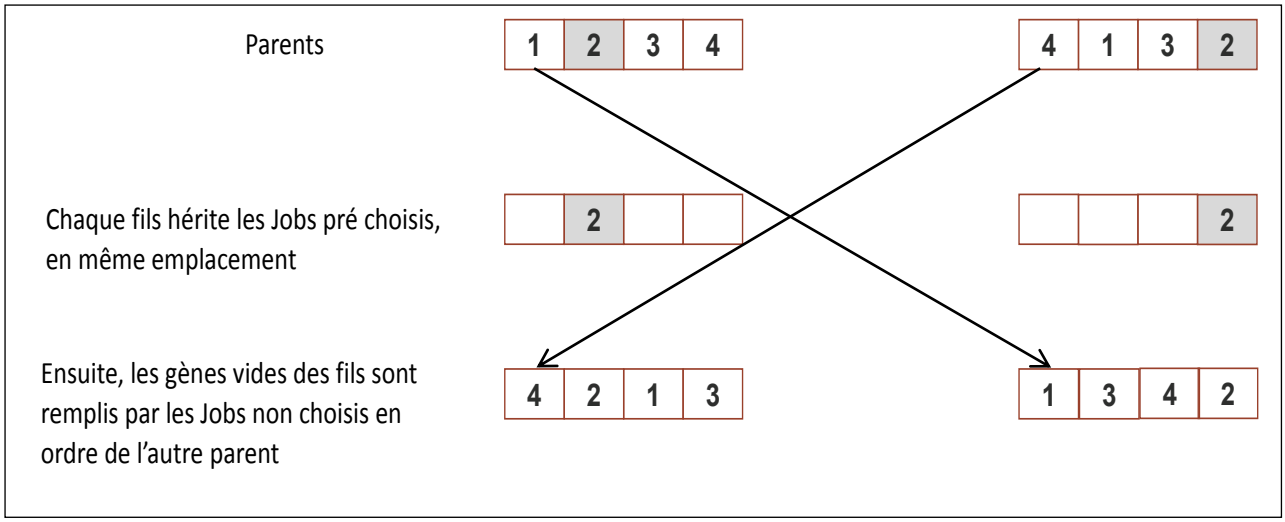


Figure III- 4-Le croisement de l'ordre des jobs

3.3.2 La probabilité de croisement Pc :

Pour la sélection des chromosomes à croiser on a basé sur la probabilité de croisement dont sa valeur reflète l'intensité des Changements appliqués sur la population. Généralement, la probabilité de croisement est comprise entre 0,5 et 0,9.

Pour notre cas on a choisir 0.8

Algorithme de croisement

Pour tous les chromosomes (ch) **faire**
Choisir les bons chromosomes (taux de croisement=0.8)
Pour deux chromosomes choisis
Fixé une valeur aléatoire (v) dans l'intervalle [0,n]
Remplacer les valeur non fixé de ch1 vers ch2 et l'inverse (ch2 vers ch1)
Fin pour
Pour taux de croisement =0.2
Choisir les mauvais chromosomes
Pour deux chromosomes choisis
Fixer une valeur aléatoire (v) dans l'intervalle [0,n]
Remplacer les valeur non fixé de ch1 vers ch2 et l'inverse (ch2 vers ch1)
Fin **Pour**
Fin Pour
Fin Pour

Algorithme- III- 2-le croisement

3.4 Mutation

Le but de cet opérateur est la diversification de l'espace de solution afin d'éviter une convergence rapide vers un seul type d'individu, cette diversification se fait par l'introduction des perturbations génétiques au niveau des individus. On distingue deux principes de Mutation : mutation par position et mutation par valeur. Pour notre cas on a choisir mutation par position.

3.4.1 Mutation de position

Le principe de la mutation de position consiste à choisir deux emplacements sur le chromosome en question, puis permuter les deux gènes. Ce processus est répété jusqu'à atteindre l'amplitude de la mutation.

L'exemple suivant (Figure III-5-) montre une mutation par permutation de positions 1 et 3 du chromosome suivant :

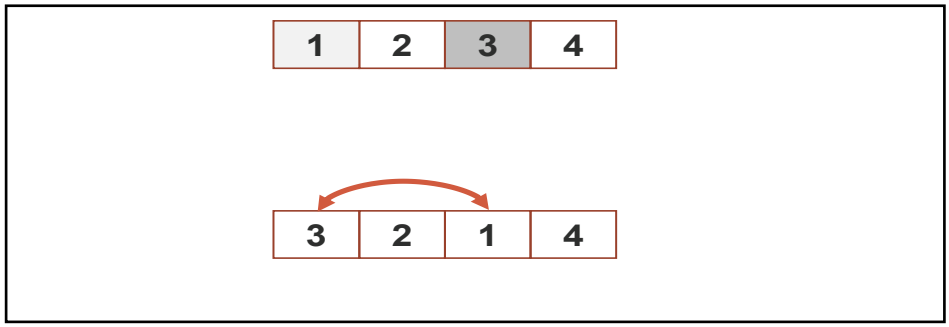


Figure III- 5-Mutation de position

3.4.2 La probabilité de mutation Pm

La probabilité de mutation c'est une valeur permet de définir le nombre de chromosomes sur lesquels on va faire la mutation. Un taux élevé de *Pm* risque de converger vers une solution sous-optimale et à la perte de la population originale, c'est pour cette raison que la probabilité de mutation est généralement faible. Pour notre cas on a choisir 0.1.

Algorithme de mutation :

```
% Un chromosome est choisi selon la probabilité Pm=0.1
Pour toute la population faire
    Si le chromosome est choisi
        Deux gènes sont choisis aléatoirement
        On inverse la position de ces deux gènes
    Fin Si
Fin Pour
```

Algorithme- III- 3-Mutation

3.5 La sélection

Le principe de la sélection consiste à choisir les individus les mieux adaptés selon leurs valeurs de fitness dans le but de former la nouvelle génération. Au cours du processus itératif (reproduction et sélection), chaque individu est sélectionné proportionnellement à sa fonction « *Fitness*», donc, plus la Fitness d'un individu est élevée, plus il aura une grande chance d'être sélectionné par rapport à un autre individu avec une fitness moins élevée. Pour notre problème on a choisir la sélection par la roue de fortune (roulette Wheel).

3.5.1 La sélection par la roue de fortune (roulette de Wheel)

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

L'évaluation permet de calculer la fitness d'un individu. En cas de problème de minimisation, la finesse croit inversement avec la fonction objectif. La fitness d'un individu xi (par exemple) est mesurée par :

$f(x_i) = \frac{1}{C(x_i)}$ où C (x_i) est la valeur de la fonction objectif obtenu avec x_i

La probabilité de sélection d'un individu x_i, notée prob(x_i), est proportionnelle à sa fitness f(xi) et telle que $\sum$ prob(x_i) = 1. Elle est définie :

$$prob(x_i) = \frac{f(x_i)}{\sum_1^n f(x_i)} \quad \text{.....1}$$

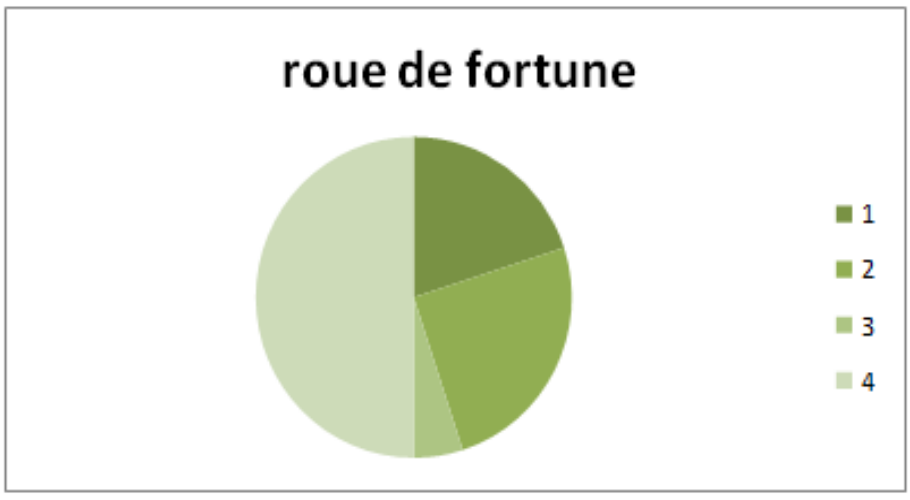


Figure III- 6-sélection par la « roue de fortune »

Algorithme de selection :

Classer les individus de la population courante selon leurs valeurs de fitness(ordre croissant)
Pour i=1 à n faire
    Calculer probabilité de selection pi
    Générer un nombre x aparetenant à l'intervalle [0,1]
    Si x<pi **alors**
        Selectionner l'individu i
    Fin si
Fin pour

Algorithme- III- 4-la Sélection

3.6 **Evaluation Des Individus**

L'évaluation des individus peut se faire selon un ou plusieurs critères, tels que : le Makespan, le retard algébrique, …etc.

Sur la base de ces critères, la *fitness* de chaque individu est estimée. Généralement, l'évaluation de fitness des chromosomes se fait en trois étapes :

1. La construction de l'ordonnancement correspondant au chromosome,
2. Le calcul de fonctions objectifs : Makespan, retard algébrique, …etc.,
3. Le calcul de fitness en fonction de la fonction objectif,

Ensuite, le tri et la sélection des chromosomes se font en fonction de leurs valeurs de fitness.

3.6.1 Calcule du Makespan

Pour calculer le Makespan (Cmax) nous avons utilisé la méthode de diagramme de Gantt qui représente l'ordonnancement des différents job à exécuter dans les machines M1,M2,M3,M4 dont les jobs (soit per exemple 1,2,3,4) ont des routages spécifiques (ex :le job 1 doivent être exécuter tout d'abord dans la machine 1 puis la machine 2 puis la machine 3 puis la machine 4) , des temps d'exécution (Processing time) et temps de transport vers chaque machine (Tableau III-1)

En respectant l'ordre d'exécution des produits le C_{max} représente le temps de la sortie du dernier produit exécuté dans le système, il est calculé par la relation suivante :

$$\mathbf{C_{max} = Max\ (\ C_k\ +\ x_j * t_{jk})} \ldots\ldots\ldots\ldots\ldots\ldots \mathbf{2}$$

t_{jk} : temps de transport des produits de la machine k vers la sortie

X_j : variable de décision = $\begin{cases} 1 & si\ j\ le\ dernier\ job\ sur\ k\ et\ k\ dernière\ machine\ traite\ j \\ 0 & sinon \end{cases}$

C_k représente le temps de disponibilité de la machine k qui égale au temps de fin d'exécution (E_j) du dernier produit déjà exécuté dans cette machine.

Pour calculer le E_j nous somme devant deux cas possibles

A). **Dj <= C_k** c-à-dire le produit est disponible et prêt d'être exécuté dans la machine k mais la machine n'est pas disponible, le produit attend jusqu'à la disponibilité de la machine pour qu'il soit traité (Figure III- 7). Dans ce cas le nouveau Ej se calcule par :

$$E_j = C_k + P_{jk} \quad \ldots\ldots\ldots\ldots\ldots\ldots 3$$

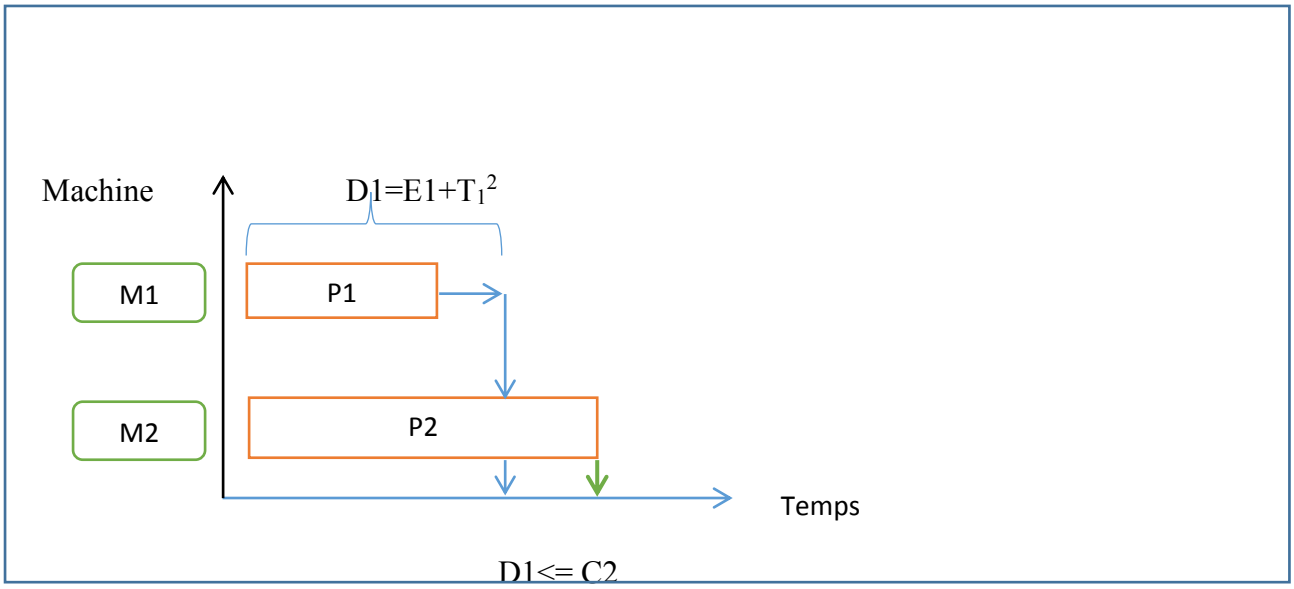


Figure III- 7-Le cas de Conflit

B). **Dj > C_k** c'est-à-dire la machine est disponible et le produit est en cours d'exécution dans une autre machine, alors le produit entrent directement dans machines après la fin d'exécution d'opération précédente (**pas de conflit)** (la figure III.8)**.** Dans ce cas le Ej sera calculé par la relation :

$$Ej = D_j + P_{jk} \quad \ldots\ldots 4$$

Dont : $D_j = E_j + t_{jk}$……5

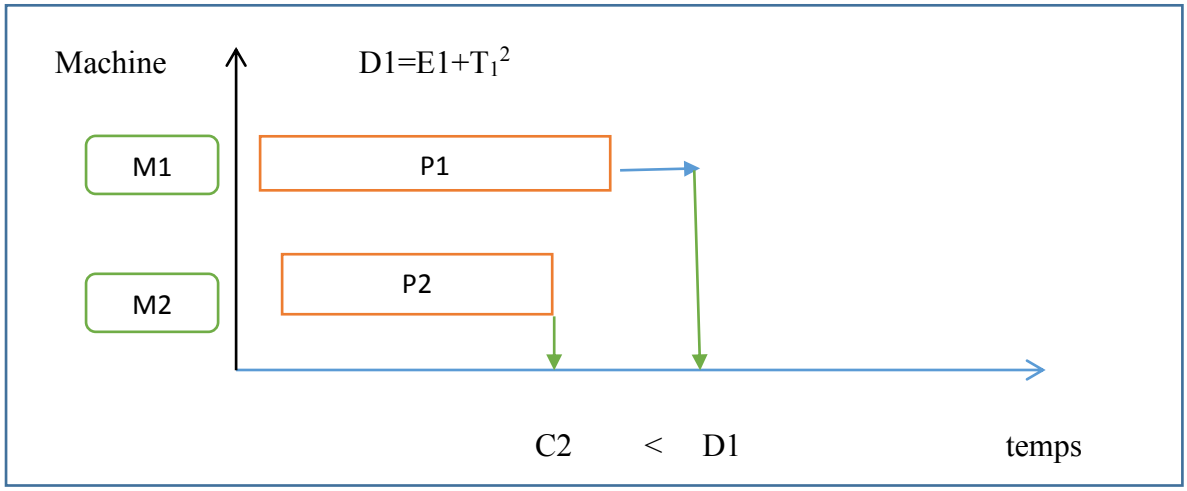


Figure III- 8-- Le cas de Pas de Conflit

En générale on peut calculer E_j comme suit :

$$E_j = \begin{cases} C_k + P_{jk} & \text{si } D_j \leq C_k \\ D_j + P_{jk} & \text{si } D_j > C_k \end{cases} \quad \text{……6}$$

Alors pour calculer le Cmax on doit toujours remplacer la valeur disponibilité de machine par la valeur de la fin d'exécution de job c'est-à-dire C_k=Ej. Enfin on ajoute les temps de sorties pour chaque machine dont le Cmax sera égale à **Max ($C_k + x_j * t_{jk}$)**

Algorithme fonctions objectifs : Makespan

Initialiser C_k, D_j, E_j

Pour k=1..4 (4 machines)

1) sélection des jobs qui doivent être exécuté sur la machine

Pour j=1..4 (4 type de produits)
Si p_j^k >0 (le job i sera exécuté dans la machine)
Si $l_j = o_j^k$ (l'opération suivante du job j est la même opération qui sera exécutée dans la machine k)

$$Dj = Ej + t_j^k$$

Fin Si
Fin Si
Fin Pour

Si les Dj sont égaux
Classer les jobs selon leur ordre dans le chromosome
Sinon
Classer les jobs selon l'ordre croissant de Dj
Fin si

2) calcul de Cmax

Pour les jobs sélectionnés

Si $Dj <= C_k$
$Ej = C_k + P_j^k$
Sinon
$Ej = D_j + P_j^k$
Fin si

$C_k = E_j$

Fin Pour

$Cmax(k) = Max\ C_k + T_j^k$

Algorithme- III- 5-le Makespan (Cmax)

4. Condition d'arrêt

Comme toute procédure itérative, un AG s'arrête si une certaine condition d'arrêt est vérifiée. Plusieurs conditions peuvent être définies, citons par exemple :

4.1 Arrêt après un nombre de générations fixé à priori

C'est la plus utilisée lorsqu'un impératif de temps de calcul est imposé ou lorsque la performance de l'algorithme est à tester. Un nombre de générations est produit signifie qu'un pourcentage de l'espace des solutions est exploré. Dans nos expérimentations ultérieures, c'est cette stratégie qui est adoptée.

4.2 Une combinaison des deux conditions

On arrête lorsqu'on évolue plus dans la qualité de la solution ou bien lorsqu'on atteint un nombre maximum de générations.

5. Simulation et résultats

Pour valider l'algorithme proposé dans la partie précédente nous avons développé un programme sur le logiciel Matlab qui est ensuite appliqué sur plusieurs exemples de petit, moyen et grand taille.

5.1 **Exemple de petite taille (problème de 4 produits)** :

Dans ce premier exemple de petite taille nous proposant de comparer les résultats obtenus par notre algorithme (en utilisant le logiciel Matlab) et sels trouvés manuellement. Les deux Tableaux suivants représentent successivement le routage et les temps opératoire de chaque Job.

	Opération(1)	Opération(2)	Opération(3)	Opération(4)
Job 1	M1	M2	M3	M4
Job 2	M2	M3	M4	/
Job 3	M1	M3	M4	/
Job 4	M1	M4	M2	M3

Tableau-III- 2routage de jobs

JOB/MACHEN	M1(temps: s)	M2(temps: s)	M3(temps: s)	M4(temps: s)
J1	40	35	80	90
J2	/	45	50	25
J3	60	/	35	20
J4	30	20	55	28

Tableau-III- 3-Temps opératoires sur les machines pour job

La simulation a était faite sur un ordinateur Acer (2 core) dont nous avons obtenu les résultats suivants :

- **Résultat obtenu avec Matlab pour la population initial**

Une population qui contient N chromosomes est une matrice de taille (N_{pop} N_{bits}) , elle peut être généré en utilisant la fonction aléatoire rand sous Matlab « **pop=round (rand (N_{pop} N_{bits}) »** pour notre cas on a N_{pop}=10 N_{bits}=4

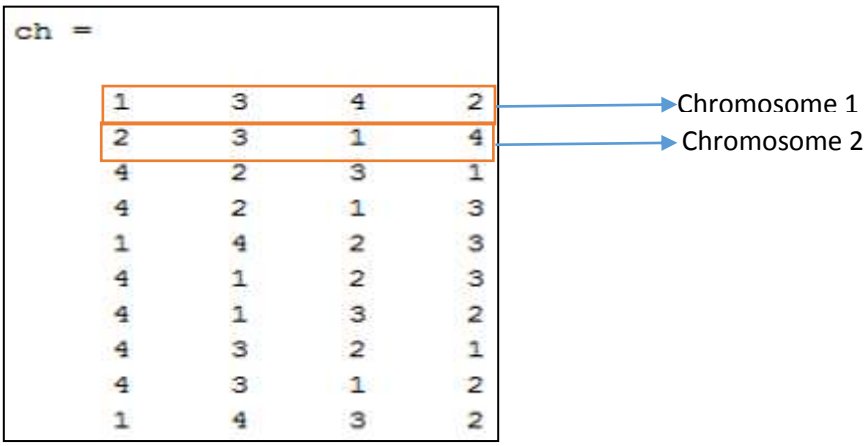


Figure III- 9-La population initiale générée

- ***Résultat obtenu avec Matlab pour la sélection :***

Après la création de population initiale sur Matlab l'algorithme réordonne les chromosomes selon la valeur de la fonction objectif (Cmax) La sélection de chromosome se fait avec une probabilité de p=0.8

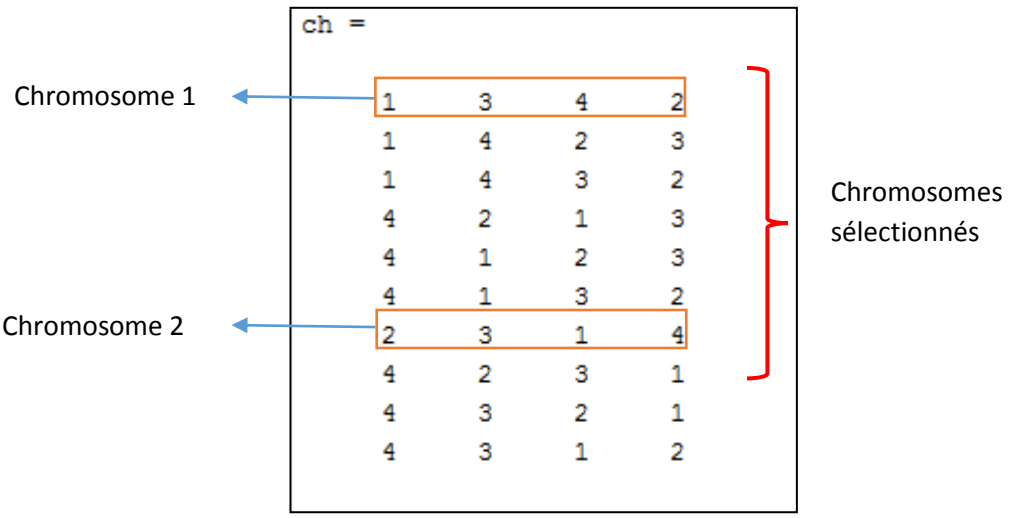


Figure III- 10-La population selon l'ordre de Cmax

- ***Résultat obtenu avec Matlab pour le croisement :***

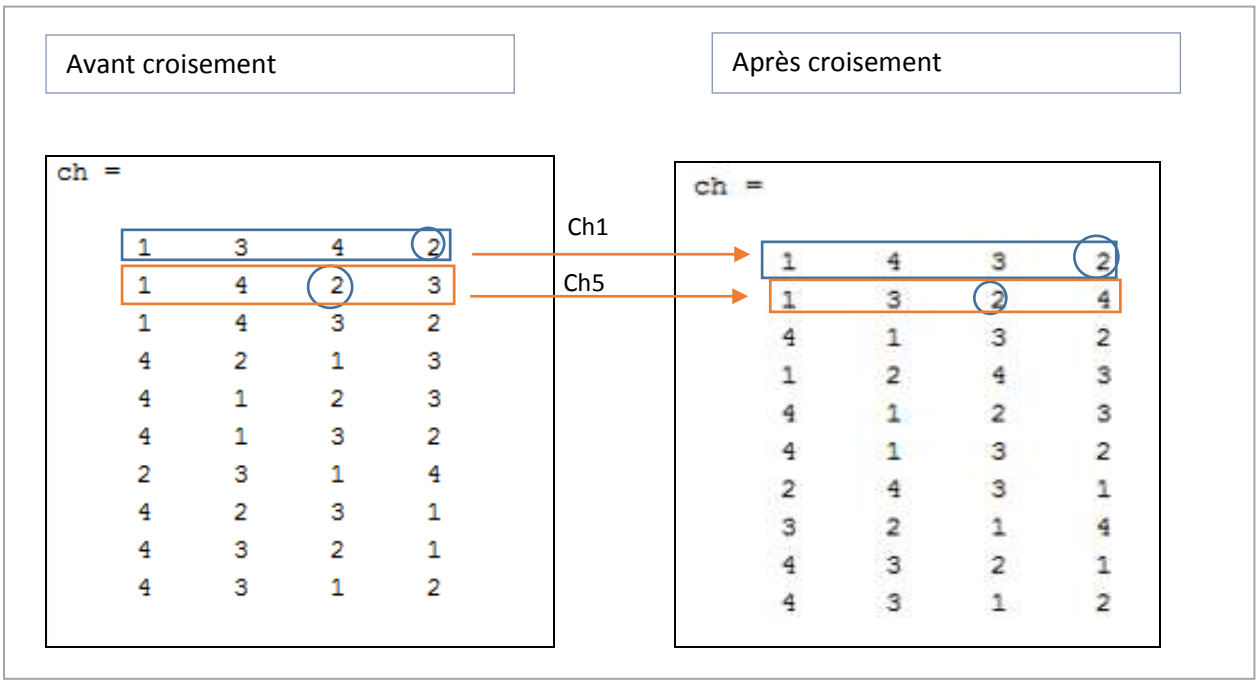


Figure III- 11-Exemple de la procédure de croisement

- **Résultat obtenu avec Matlab pour la mutation *:***

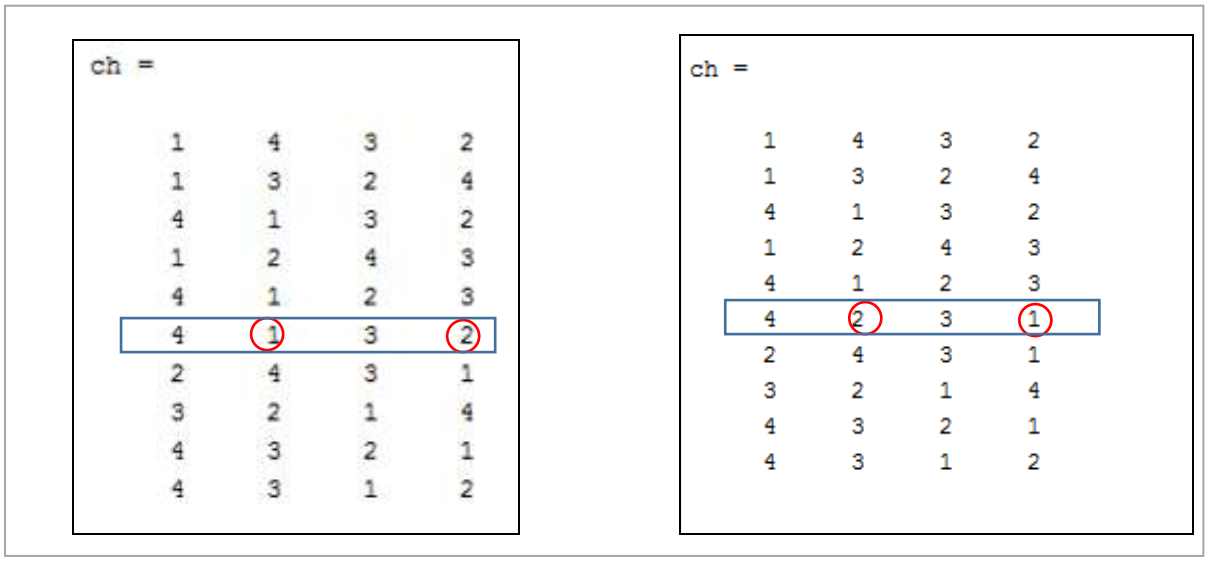


Figure III- 12-Exemple de la procédure de mutation

- ***Résultat obtenu avec Matlab pour fonction objectif (Cmax)***

La figure suivante montre le résultat de Cmax obtenu pour le chromosome ch : 3, 1, 4, 2

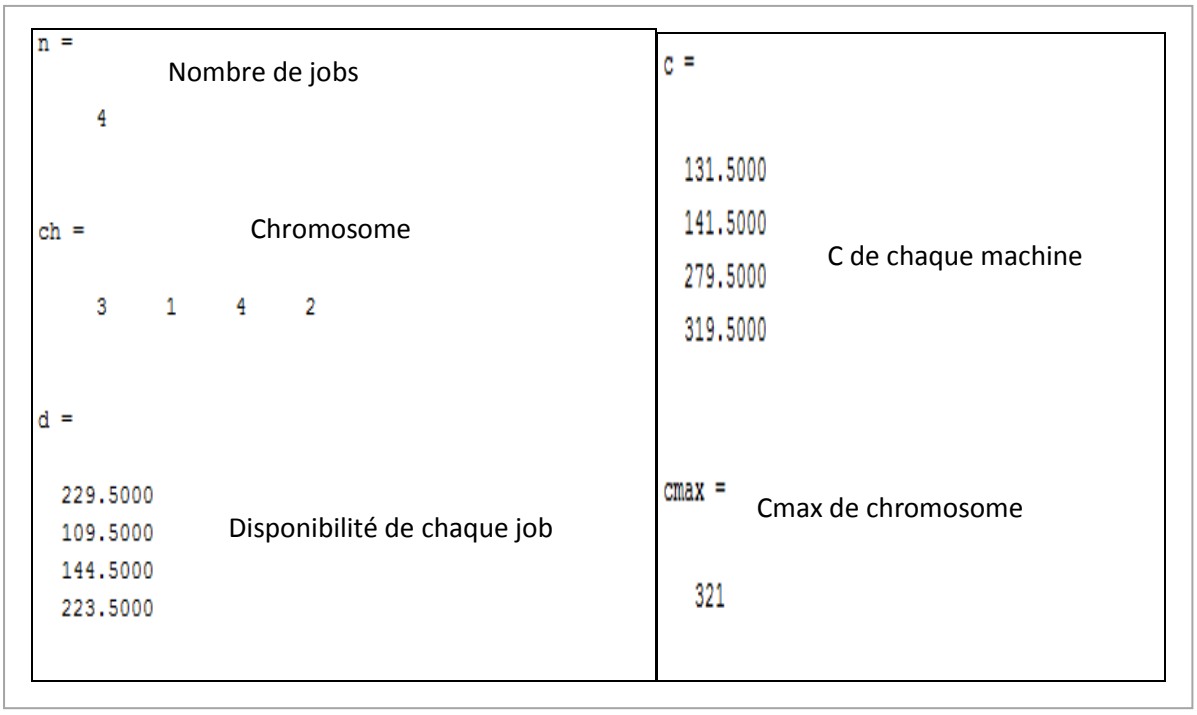


Figure III- 13-Exemple du calcul de Cmax

- ***Résultat obtenu avec Matlab pour La sélection par la roue de fortune :***

Sélectionner le meilleur chromosome parmi les chromosomes mutés

```
ans =

   301


indiop =

     1


solopt =

     1     4     3     2
```

Le Cmax minimum

Position de chromosome sélectionné

Figure III- 14-Exemple du résultat de sélection

- ***Résultat final obtenu avec Matlab après sept itérations :***

Après sept itérations l'évolution du programme nous avons obtenu les résultats suivants :

```
cmaxit =

   301   301   301   301   301   301   301
```

Le Cmax de chaque itération

```
chopt =

     1     4     3     2
     1     3     4     2
     1     3     2     4
     1     4     3     2
     1     4     3     2
     1     4     2     3
     1     4     2     3
```

Les meilleurs chromosomes pour chaque itération

Figure III- 15-Résultat finale après 7 itérations

- ***Résultat final obtenu avec Matlab représenté par le diagramme de Gantt :***

Le diagramme présenté dans la figure-III-16 représente le diagramme de Gantt d'une solution parmit les meilleures solutions obtenu après l'application de notre algorithme sur le problème 4x4

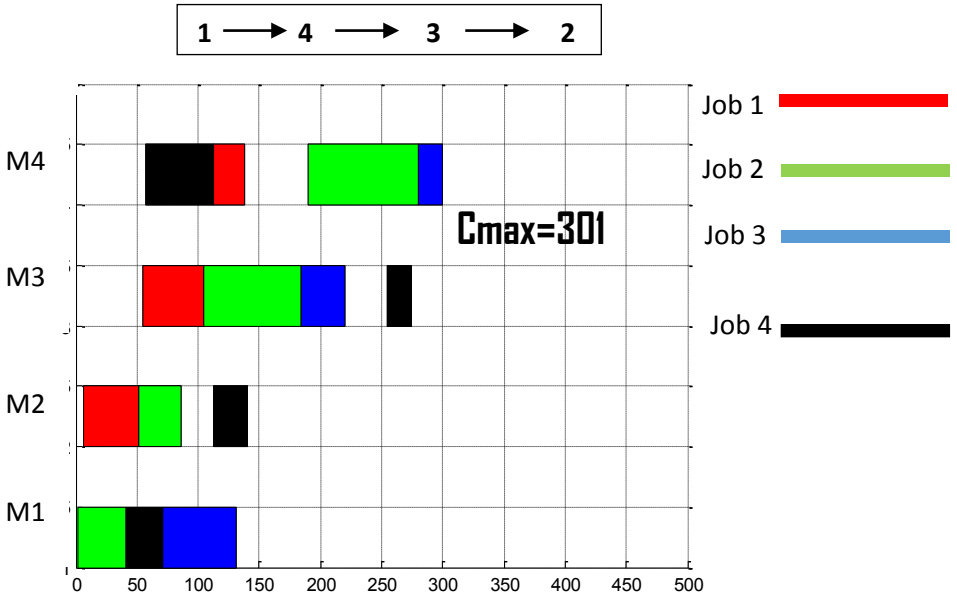


Figure III- 16-Exemple de Diagramme de Gant obtenu par Matlab

Après plusieurs simulations du programme avec toujours les même données et le même nombre des itérations nous avons recensé les résultats montrés dans la figure III.17

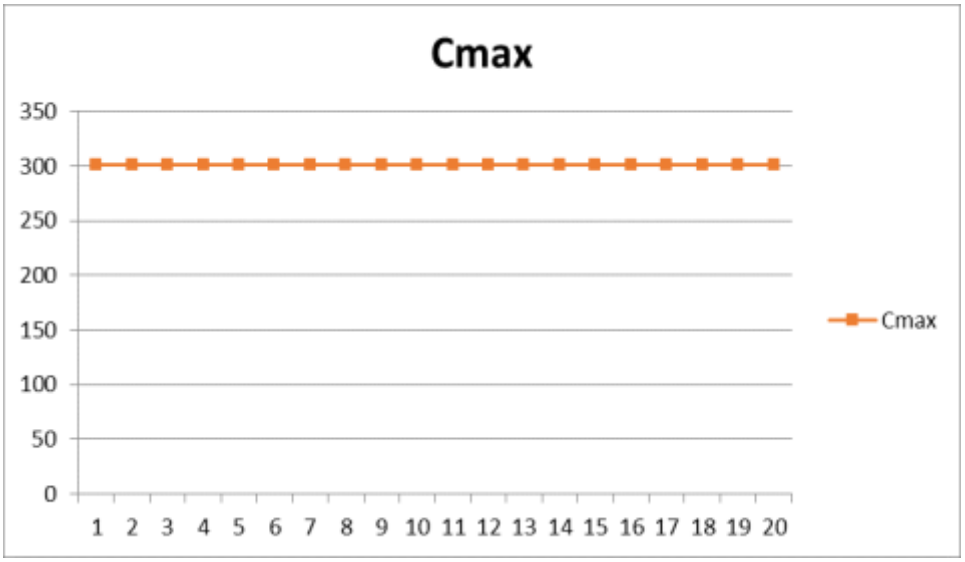


Figure III- 17-Résultat obtenu par notre algorithme après 20 tests

Pour évaluer ces résultats nous avons calculé le Cmax manuellement pour tous les combinaisons possibles (pour cet exemple 4 produits il y a 24 combinaisons donc 24 solutions possibles) et nous avons trouvé les résultats représentés dans le tableau suivant

solu	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Cmax	301	301	351	311	351	301	301	351	351	351	311	311
Solu	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**
Cmax	321	351	321	311	301	351	351	301	301	301	321	321

Tableau-III- 4-Résultat obtenu manuellement pour 24 solutions possibles

A travers ce résultat nous remarquons que la solution optimal Cmax égale à 301s cette solution est la même trouvée par notre algorithme. Alors nous pouvons dire que pour les problèmes de petites tailles (comme est le cas de notre exemple) l'algorithme que nous avons développé donne des solutions à 100% optimaux**.**
L'exemple suivant montre la procédure comment nous avons pu calculer le Makspen manuellement

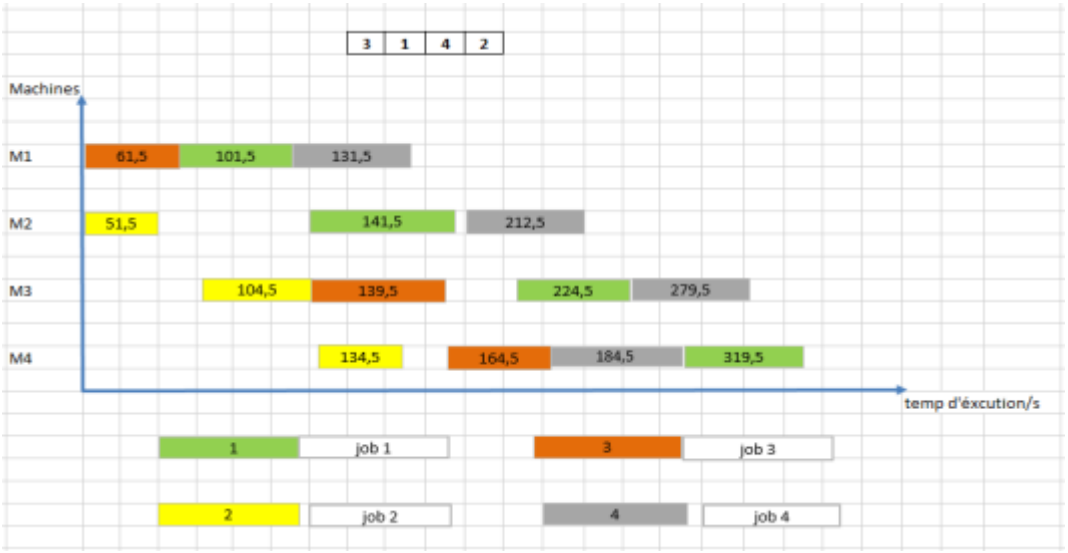


Figure III- 18-diagramme de Gantt (chromosome 1->2->3->4)

En cas de D1 > C2 donc E1=D1+ P_1^2 **(Pas de conflit)**

En cas de D3 <=C3 donc E3=C_3+ P_3^3 **(avec conflit)**

Calcule de Cmax :

C(1)= 319.5+1.5= 321 ; C(2)=134 .5+1.5=136 ; C(3)= 184.5+1.5= 186 ; C(4)=279.5+6.5=286

Donc le **Cmax = 321 s**

5.2 **Exemple de moyen taille (problème de 6 produits)**

Dans cet exemple nous avons essayé d'appliquer notre approche de résolution sur six produits supposés d'être exécutés dans notre système. Les donné montrées dans le tableau III.5 et le tableau III.6 qui représente successivement le routage et le temps d'exécution des produits sont inspirées de la littérature

	Opération(1)	Opération(2)	Opération(3)	Opération(4)
Job 1	M1	M2	M3	M4
Job 2	M2	M3	M4	/
Job 3	M1	M3	M4	/
Job 4	M1	M4	M2	M3
Job 5	M1	M2	M3	M4
Job 6	M2	M3	M4	/

Tableau-III- 5-les machines utilisées pour chaque produit

	Machine 1	Machine 2	Machine 3	Machine 4
Produit 1	40	35	80	90
Produit 2	0	45	50	25
Produit 3	60	0	35	20
Produit 4	30	20	55	28
Produit 5	41	33	80	90
Produit 6	0	43	55	25

Tableau-III- 6-Routage de Produits

Nous avons supposé que la taille de population est de 20 chromosomes et le nombre d'itération est de 7 et nous avons obtenu les résultats suivant :

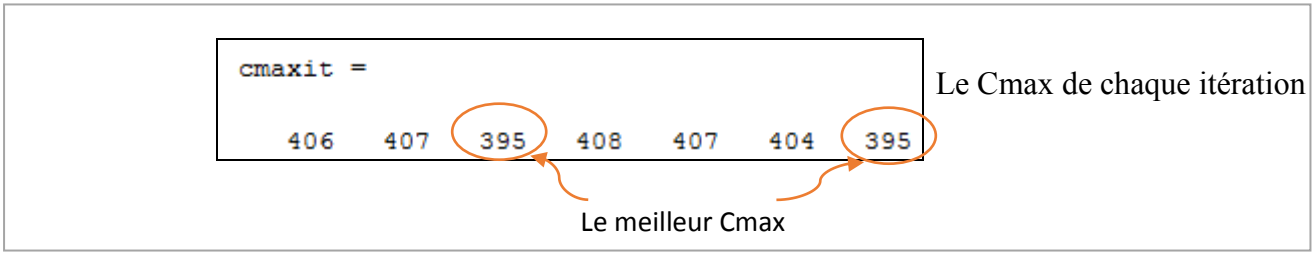


Figure III- 19-Résultat obtenu par Matlab après 7 itérations

Test	1	2	3	4	5	6	7	8	9	10	11	12
Cmax de la Meilleure solution	395	412	456	506	417	395	500	455	438	438	500	412

Tableau-III- 7-Résultat obtenu manuellement pour 12solutions possibles

Nous avons remarquée, à travers cet exemple, que le pourcentage de tomber dans les meilleures solutions est de 80%. Ce pourcentage élevé de la sélection du meilleure chromosome nous permet de dire que l'algorithme proposé donne des meilleures solutions dans les problèmes de moyen tailles.

La figure suivante présente le diagramme de Gantt d'une des meilleures solutions trouvées.

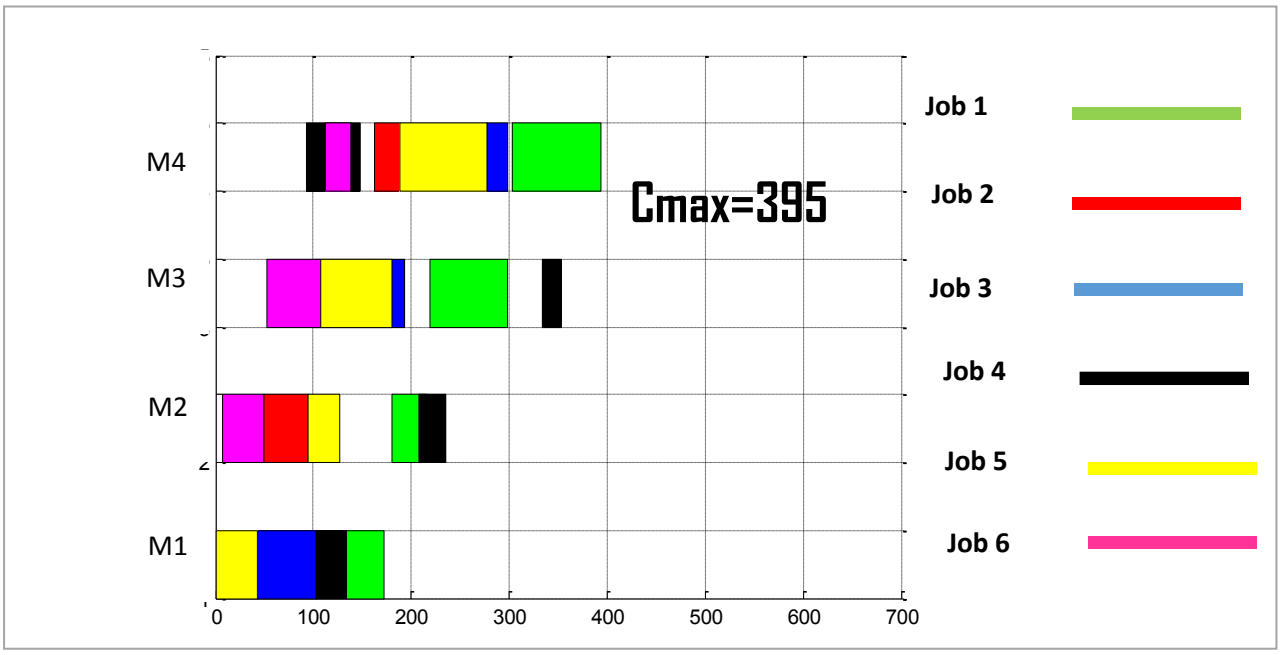


Figure III- 20-Diagramme de Gant de 6 produits

5.3 **Exemple de moyen taille** :

Dans cette partie nous avons choisir, pour complété l'évaluation de notre programme proposé, des problèmes de grandes tailles

Nous supposons que la taille de population est de 10 chromosomes et le nombre d'itération est de 10 itérations.

a. Problème de quinze produits

le tableau III- 9 représente les temps d'excution et le tableau III- 10 représente le routage de produits

Produit/Machine	M1	M2	M3	M4
1	40	35	80	90
2	0	45	50	25
3	60	0	35	20
4	30	20	55	28
5	20	0	0	10
6	40	35	80	90
7	46	35	81	92
8	44	35	87	90
9	40	35	80	90
10	46	35	81	92
11	44	35	87	90
12	44	35	87	90
13	40	35	80	90
14	46	35	81	92
15	44	35	87	90

Tableau-III- 8-Les Temps D'opération Des Produits

Produit/Opération	O1	O2	O3	O4
1	1	2	3	4
2	2	3	4	/
3	1	3	4	/
4	1	2	3	4
5	1	4	/	/
6	1	2	3	4
7	1	2	3	4
8	1	2	3	4
9	1	2	3	4
10	1	2	3	4
11	1	2	3	4
12	1	2	3	4
13	1	2	3	4
14	1	2	3	4
15	1	2	3	4

Tableau-III- 9-Routage De Produits

Après simulation nous avons obtenu les résultats montré dans la figure suivant :

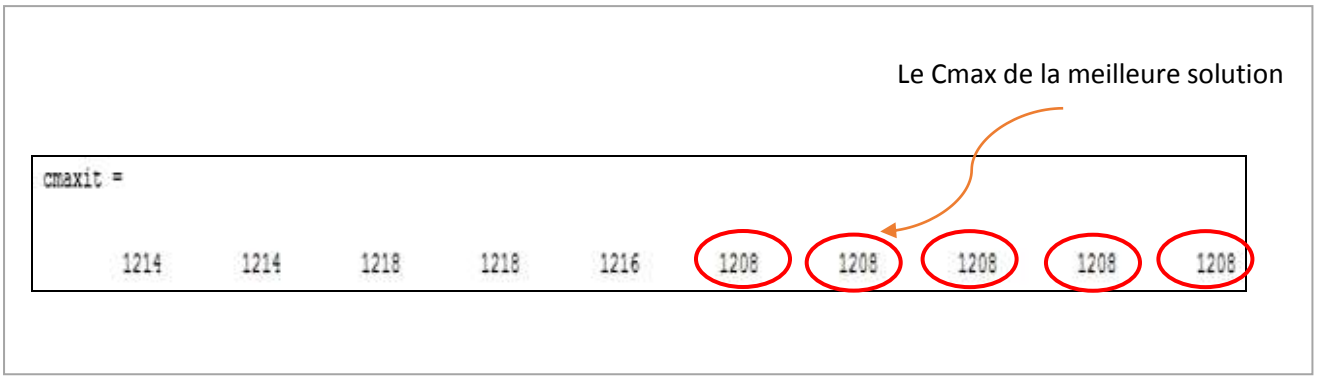


Figure III- 21-Résultat obtenu avec Matlab après 10 itérations

Cette figure monter la variation de la valeur de Cmax à travers les différentes itérations faites par le programme. Nous remarquons que la probabilité de tombé dans la meilleure solution est presque de 50%.

b. **Problème de** trente produits :

Pour ce problème, inspéré de la litérature, les temps d'excution et le routage des produits dans le système sont présenté dans le tableaux III- 10 suivant :

Produit /Machine	1	2	3	4
1	99	43	6	99
2	19	24	65	16
3	54	62	93	78
4	60	16	79	84
5	49	52	46	50
6	59	65	85	40
7	10	7	22	36
8	53	74	93	26
9	76	72	42	17
10	18	79	93	71
11	52	61	45	60
12	63	73	82	84
13	90	90	77	33
14	87	47	58	34
15	35	38	30	93
16	18	92	62	59
17	2	26	17	18
18	6	97	65	58
19	31	79	87	79
20	96	1	75	42
21	84	82	16	61
22	46	29	50	12
23	10	39	49	56

24	70	63	68	97
25	81	97	65	60
26	39	6	59	34
27	98	85	51	9
28	59	68	3	8
29	92	21	53	64
30	51	70	94	80

Tableau-III- 11-Les Temps D'opération Des Produits

Après la simulation de 10 itérations nous avons obtenu les résultats suivant :

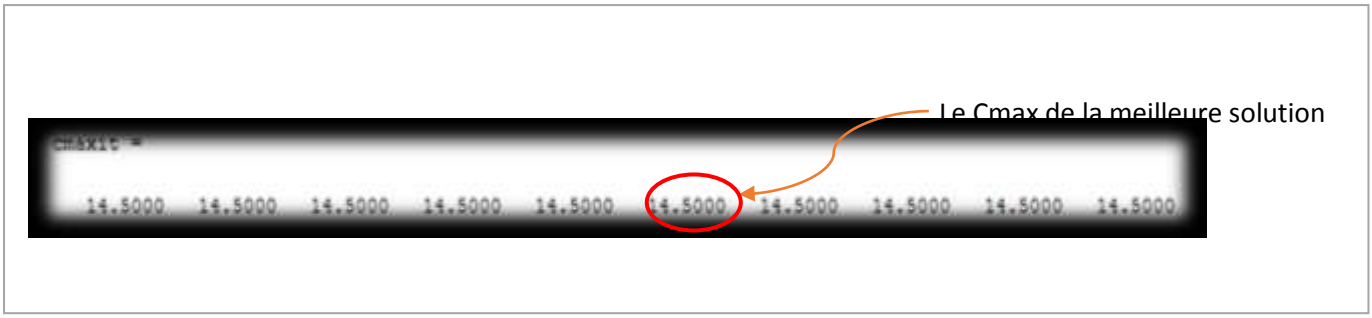


Figure III- 22-Résultat obtenu avec Matlab après 10 itérations

Pour ce problème nous avons remarqué que le programme nous permet de trouver la meilleure solution avec un pourcentage presque de 100%.

Dans cette partie nous avons remarquée après l'adaptation notre algorithme sur des problèmes de tailles différentes que le résultat obtenus sont très motivants.

6. Conclusion

Au travers ce chapitre nous avons présenté et expliqué l'implémentation de l'algorithme que nous avons proposé pour la résolution du problème de job shop avec contrainte de transport où les produits sont transportés dans un sens unidirectionnel. Cet algorithme est développé à la base de la méthode algorithmes génétiques qui nous donne des résultats plus proche de l'optimum dans des temps de traitement très faibles. La simulation a était faite, en utilisant le logiciel Matlab, sur plusieurs exemples de différents tailles (petite, moyen et grande) à travers lesquels nous avons essayé de tester l'efficacité et la performance de notre algorithme

A partir des différents résultats obtenus nous avons remarqué que notre algorithme est très efficace et donne des meilleures solutions surtout pour les problèmes de petites tailles où il donne des solutions presque 100% optimums.

CONCLUSION GENERALE

Conclusion générale

Les problèmes de l'ordonnancement sont présents dans tous les secteurs de l'économie et constituent une fonction importante en gestion de production. Un problème d'ordonnancement consiste à allouer dans le temps des tâches à des ressources qui existent en quantité limitée, tout en satisfaisant un ensemble de contraintes.

Le problème d'ordonnancement des ateliers de type Job Shop est l'un des problèmes d'ordonnancement intensivement étudiés. C'est un problème extrêmement complexe. Il est classé parmi les problèmes combinatoires difficiles au sens fort. Cette complexité est due à l'explosion combinatoire du nombre de solutions qui croît exponentiellement avec la taille du problème. L'utilisation de 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 Métaheuristiques comme une approche très prometteuse.

Dans ce travail, nous avons traité une problématique d'ordonnancement de type Job Shop avec contrainte de transport dont nous avons fait le recours aux algorithms génétiques pour la résolution de cette problématque. Un cadrage théorique de notre thème a consisté à présenter les problèmes de l'ordonnancement en général, le problème d'ordonnancement des ateliers de type Job Shop. La deuxième partie du travail consisté à présenter les différentes méthodes de résolution d'un problème d'ordonnancement de job shop La méthodes retenue est les Algorithmes Génétiques qui permet de rechercher le meilleur ordre d'exécution des opérations au niveau de chaque machine. La troisième partie consiste à présenter la problématique et l'algorithm proposé qui se bas sur le principe des Algorithms génétques .

La simulation du programme développé a était faite, en utilisant le logiciel Matlab, sur plusieurs exemples de différents tailles (petite, moyen et grande) à travers lesquels nous avons essayé de tester l'efficacité et la performance de notre algorithme.

A partir des différents résultats obtenus nous avons remarqué que notre algorithme est très efficace et donne des meilleures solutions surtout pour les problèmes de petites tailles où il donne des solutions presque 100% optimums.

Pour terminer notons que le problème de job shop généralise d'autre problème d'ordonnancement classique comme le problème à une machine ,le flow shop de permutation,

et le flow shop générale ainsi que l'algorithme qui nous proposé de traite le problème de job shop peut résoudre des instance des ces particulière.

Perspective

Dans notre travail nous n'avons pas pu traiter le problème dans tous les cas possibles, il reste autres cas non pas été considérés ici notamment le cas de blocage des produits alors une extension de notre programme sera nécessaire pour généraliser la résolution de notre problématique.

REFERENCES BIBLIOGRAPHIQUES

[1]:RG ASKIN, CR STANDRIDGE, "modeling and analysis of manufacturing Systeme" , John wiley and Sons, 1993
[2] : « Projet fin d'étude simulation d'une chaine flexible d'assemblage avec ARENA, M^{elle} BENDAHMANE Zhor1999-2000 »
[3] : Giard V.,« Gestion de la production », Economica, 2003
[4] :Artigues C., « Ordonnancement En Temps Réel d'Ateliers avec temps de Préparation des ressources », Thèse de doctorat, Université Paul Sabatier, Toulouse, 1997
[5] :Tamani K., « Développement d'une méthodologie de pilotage intelligent par régulation de flux adaptée aux systèmes de production», Thèse de doctorat, Université de SAVOIE.
[6] : Giard V., « Gestion de production», 2 éme édition, Economica, paris, 1988.
[7] : Mebarek K., « Utilisation des stratégies Méta-heuristiques pourL'ordonnancement des ateliers de type Job Shop», Mémoire de Magister, Université de Batna, Algérie2008.
[8]: VACHER J., « 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, Université du Havre, 2000.
[9] : Blondel F., « Gestion de la production», 3 ^{éme} édition, DUNOD, Paris, 2000.
[10] : Javel G., « Organisation et gestion de la production, cours avec exercices corrigés 4 éme édition, Dunod, Paris, 2010.
[11]:Carlier, J. et Chrétienne, P. (1988), Problèmes d'ordonnancement : modélisation / complexité /algorithmes. Edition Masson.
[12]: M. Pinedo, « Scheduling : Theory, Algorithms and systems ». Prentice-Hall, Englewood Clis, New Jersey, 1955.
[13]: J. Blazewicz, W. Domschke E. Pesch (1996), The job Shop Scheduling Problem :Conventionnal and new solution techniques. European Journal of Operational Research.
[14] :F.A. Rodammer et K. Preston White, « A recent survey of production scheduling ». IEEE Transaction on Systems, Man and Cybernetics, pp. 6-18, 1999.
[15] J. Carlier et P. Chrétienne, « Problèmes d'ordonnancement, ModélisationComplexité, Algorithmes ». Edition Masson, Paris, 1988.
[16]: Lopez P. & Esquirol P .Lordonnancement, Economica, Parise 1999.
[17]: F. Farhoodi, "A knowledge-based approach to dynamic job shop scheduling", International Journal of computer Integrated Manufacturing, vol.3, n2, 1990, pp.84-95.
[18]:B Grabot, "objective satisfaction assesment using neural nets for balancing multiple objectives", International Journal of production research, vol.36, n9, 1998, pp. 2377-2395.
[19]: Baptiste P. (1998), une étude théorique et expérimentale de la propagation des contraintes de ressources. Thèse de doctorat, université Technologique de Compiègne.
[20]: Letouzey A. (2001), ordonnancement interactif basé sur des indicateurs : Application à la gestion de commandes incertaines et à l'affectation des opérateurs. Thèse de doctorat, Institut National Polytechnique de Toulouse.
[21]:Fisher H. e Thompson G.L. (1963), Probabilistic learing combination of local job-shop scheduling rules. Industrial Scheduling, Prentice Hall;: 225-251.

[22]: J. Blazewicz, W. Domschke E. Pesch (1996), The job Shop Scheduling Problem :Conventionnal and new solution techniques. European Journal of Operational Research.
[23]: thèse de doctorat de Mohand LARABI à Université Blaise Pascal - Clermont Ferrand II, Spécialité : INFORMATIQUE thème Le problème de job-shop avec transport : modélisation et optimisation, 15 décembre 2010.
[24]: Jain A.S. and Meeran S. (1999), Deterministic job-shop scheduling: Past, present and future. European Journal of Operational Research ;113 (2): 390-434.
[25]:Manne A.S. (1960), On the Job Shop scheduling problem. Operation Research.

[26]:Brooks, G.H. and White, C.R., An algorithm for finding optimal or near optimal solution to the production scheduling problem. J. Ind. Eng; 16(l): 34440.
[27]: Greenberg H. (1968), A branch-and-bound solution to the general scheduling problem, Operation. Research; 16: 353-361.
[28]:Fisher M.L. (1973), Optimal solution of scheduling problems using Lagrange multipliers: part I. Operation Research; 21: 1114-1127.
[29]:Fisher M.L. (1973), Optimal solution of scheduling problems using Lagrange multipliers: part II. Symposium on the Theory of Scheduling and its applications.Springer.
[30]: McMahon G. and Florian. (1975), On scheduling with ready times and due dates to minimize maximum lateness. Operational Research; 23(3): 475-482.Carlier et Pinson (1989): Carlier J., E. Pinson. (1989), A Branch and Bound Method for Solving the Job Shop Problem. Management Science;:164-176.
[31]:Carlier J., E. Pinson. (1989), A Branch and Bound Method for Solving the Job Shop Problem. Management Science;:164-176
[32]: Duvivier D. Etude de l'hybridation des méta- heuristiques, Application à un problème d'ordonnancement type de Job Shop, Thèse de doctorat, Université du Littoral Côte d'Opale, LIL Calais 2000.
[33]Billaut, J.C., Vignier A., Proust C. and Portmann, M.C.: "A Survey of Hybrid flow Shop Problems". ODIP, Aussois, France.(2000).
[34]Vignier A., Billaut, J.C. and Proust, C.: "Scheduling problems type hybrid flow-shop: State of the art." RAIRO.Rechercheopérationnelle, 33(2),pp 117-183. (1999)
[35]:E. G. Negenman, « Local search algorithms for the multiprocessor flow-shop scheduling problem ». European Journal of Operational Research, vol. 128, pp. 147-158, 2001.
[36]:J. Breit, « A polynomial-time approximation scheme for the two-machine flow- shop scheduling problem with an availability constraint ». Computers & Operations Research, vol. 33, pp. 2143–2153, 2006.
[37]: R. Ruiz, C. Marotoet J. Alcaraz, « Two new robust genetic algorithms for the flow-shop scheduling problem ». Omega, vol. 34, pp. 461 – 476, 2006.
[38]: Jeffrey W .Herrman, & al. Handbaook Of Production Scheduling, Springer NY, 2006.
[39]:J. Carlier, P. Chrétienne et C. Girault, « Modelling scheduling problems with Petri nets. Advanced studies in Petri nets ».Lecture notes in Computer Science, Springer Verlag, Paris.
[40]:B. Roy, « Algèbre moderne et théorie des graphes, volume 2 ». Editions Dunod, Paris.
[41]:Gotha, « Les problèmes d'ordonnancement ». RAIRO-Recherche Opérationnelle.
[42]: A.S. Jain et S. Meeran, « Deterministic job-shop scheduling : past, present and future ». European Journal of OperationalResearch, vol. 113, pp. 390-434, 1999.
[43]:P. Chrétienne, « Les réseaux de pétri temporises ». Thèse de Doctorat, Université de Paris VI, Paris, 1983.
[44] « Projet fin d'étude étude du probleme de job shop avec un convoyeur» 3mai 2007 RUHLMANN CARINE
[45] Mémoire de Magister «Modélisation et Ordonnancement temps réel d'un Job shop» décembre 2011 HOUBAD YAMINA.
[46] Z. Kaddouri / Thèse en Informatique / 2014 / Faculté des Sciences Rabat.
[47] Lopez, P. Roubellat, Ordonnancement de la production. Paris Hermès.
[48] C. Flipo-Dhaenens, Optimisation d'un réseau de production et dedistribution, Thèse de doctorat, INPG de Grenoble. 1998, p 197.
[49] Vincent Giard, Gestion de la production et des Flux, Paris Economica.
[50] Jourdan, L., (2010). Métaheuristiques Coopératives: du déterministe au stochastique, Habilitation à Diriger des Recherches, Université de Lille I.
[52] J.-KHao,P.Galinier and M.Habib. Méthaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes. Revue d'Intelligence Artificielle Vol.

[53] M.Birrttari, L. Paquete, T. Stutzle and K. Varrentrap . Classification of metaheuristics and Design of Experiments for the analysis of component. Technical Repport AIDA -01-05. November 2001.

[55] F. Glover. tabu search fundamentals and uses. supported in part by the National Science and Engineering Council of Canada under Grants 5-83998 and 5-84181; juin 1995

[56]M. Dorigo and T. Stutzle. The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances. Technical Report IRIDIA-2000.. To appear in Metaheuristics Handbook, F. Glover and G. Kochenberger (Eds.), International Series in Operations Research and Management Science, Kluwer, 2001.

[54] D.Goldberg, Genetic algorithms in search optimization and machinelearning. Addison-Wesley, 1989.

[57] Randly L. Haupt & Sue Ellen Haupt, « Pratical genetic algorithms, Second Edition. John Wiley & Sons,Inc. New Jersey, 2004.

[59] : Fontanili F. « Intégration d'outils de simulation et d'optimisation », thèse de doctorat, Université Paris хш, 1999.

[60] Mesghouni K., « Application des algorithmes évolutionnistes dans les problèmes d'optimisation en ordonnancement de la production », Thèse de Doctorat, Université des Sciences et Technologies de Lille 1, 1999.

[61] Hao J.K., P. Galinier, M. Habib, « Méta heuristiques pour l'optimisation combinatoire et l'affectation sous contraintes », Revue d'Intelligence Artificielle, Vol.

[62] Kebabla Mebarek, « Utilisation des stratégies Méthaheuristiques pour l'ordonnancement des ateliers de type Job Shop », Thèse de magister, Université de Batna, 2008.

[58]Drummond. M Bresina. J & Swanson. K. « Just-In-Case Scheduling ».Twelfth National Conference on Artificial Intelligence (AAAI-1994),Seattle, USA, 1994.

[51]Kacem I., « Ordonnancement multicritère des job-shop flexibles : formulation, bornes inférieures et approche évolutionniste coopérative »,Thèse de Doctorat, Université de Lille 1, 2003.

[56] F. Glover, Tabu search, part I. ORSA. Journal of Computing, vol.1, pp. 190-206, 1989.

[63] E. A. Feigenbaum and J. Feldman. (Edirors). Computers and thought. McGraw-Hill Inc.

[64]. J. R. Slagle. Artificial intelligence: The heuristic p rogramming approach. McGraw-Hill. pp. 3. New York, 1971.

[65] Newella. The heuristic of George Polya and its relation to artificial intelligence. A paper given at The International Symposium on the Methods of Heuristic. University of Bern, Switzerland, Sept. 15-18, pp.16. (Published in Groner et al. (1983), pp. 195-244. 1980.

[69] 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, Université du Havre, 2000.

[67]. J. Pearl. Heuristics: intelligent search strategies for computer problem solving. pp. 3. Addison-Wesley Publ. Co, London, 1984.

[66]. [R. L. Solso. Cognitive psychology. Harcourt Brace Jovanovich, Inc.,

[68] I.H. Osman, G. Laporte. Metaheuristics: A bibliography. Ann.

[71] Javel G., Organisation et Gestion de la Production, 3eme édition DUNOD, Paris 2004

[70] Giard V., Gestion de la Production, 2ème édition, Economica, Paris, 1988.

,

ANNEXE

ANNEXE

Algorithme de colonies de fourmis de base : the « Ant System »

a- L'équation probabilité de fourmi k de une ville i choisit d'aller à la ville j

$$p_{ij}^{k}(t) = \frac{[\tau_{ij}(t)]^{\alpha} * [\eta_{ij}]^{\beta}}{\sum_{le\,N_i^k} [\tau_{il}(t)]^{\alpha} * [\eta_{il}]^{\beta}} \qquad \text{if } j \in N_i^k \qquad \textbf{(1)}$$

Où $\eta_{ij} = 1/d_{ij}$ est l'information heuristique a priori disponible, et N_i^k est l'ensemble de villes que la fourmi k n'a pas encore visité, cet ensemble forme le voisinage faisable de la fourmi k. A et B sont deux paramètres qui déterminent le degré d'influence de la densité de phéromone et de l'information heuristique sur le choix de la prochaine ville.

b- L'équation de : Déposer une piste Δ (t) sur le trajet conformément

$$\tau_{ij}(t+1) = (1-p) * \tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}(t) \quad \forall(i,j) \qquad \textbf{(2)}$$

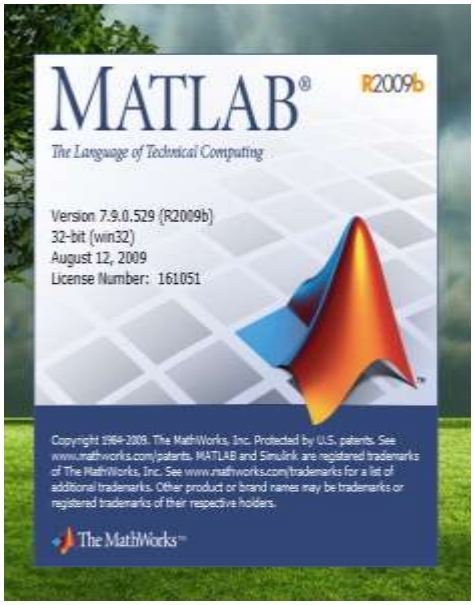
Où 0 < P = 1 est le taux d'évaporation de la phéromone et le m est le nombre de fourmis.

c- L'équation de : évaporer les pistes

$$\Delta\tau_{ij}(t) = \begin{cases} 1/L^{k}(t) & \text{If arc (i,j) is used by ant k} \\ 0 & \text{otherwise} \end{cases} \qquad \textbf{(3)}$$

Où L k (t) est la longueur de la k^{eme} excursion de la fourmi.

MATLAB (« *matrix laboratory* ») est un langage de programmation de quatrième génération et un environnement de développement ; il est utilisé à des fins de calcul numérique. Développé par la société The MathWorks, MATLAB permet de manipuler des matrices, d'afficher des courbes et des données, de mettre en œuvre des algorithmes, de créer des interfaces utilisateurs, et peut s'interfacer avec d'autres langages comme le C, C++, Java, et Fortran. Les utilisateurs de MATLAB (environ un million en 2004) sont de milieux très différents comme l'ingénierie, les sciences et l'économie dans un contexte aussi bien industriel que pour la recherche. Matlab peut s'utiliser seul ou bien avec des *toolbox* (« boîte à outils »).



La sélection par la roue de fortune :

1- **Exemple de quatre produits :**

Les meilleurs chromosomes pour chaque itération

```
chopt =

     1     4     3     2
     1     3     4     2
     1     3     2     4
     1     4     3     2
     1     4     3     2
     1     4     2     3
     1     4     2     3
```

2- **Exemple de six produits :**

D'après ces résultats en résulte le meilleure chromosome de chaque itération :

```
chopt =

    4     2     6     5     3     1
    4     2     5     6     3     1
    2     5     3     4     1     6
    3     2     6     5     1     4
    2     4     5     6     3     1
    2     4     6     5     3     1
    2     5     3     1     4     6
```

Le meilleur chromosome de l'itération i=1

Les meilleurs chromosomes pour 7 itérations

3- **Exemple de quinze produits :**

D'après ces résultats en résulte le meilleur chromosome de chaque itération :

```
chopt =

    11    10     4     6    15     5     8    14     7    12     2    13     1     9     3
    11    10     4     6    15     5     8    14     3     7    12     2    13     1     9
     5    11    10     4     6    15     8    14     3     7    12     2    13     1     9
     5    11    10     4     6    12    15     8    14     3     7     2    13     1     9
     1     9     7     4    13     5     6     3    14     2    11    10     8    12    15
     7     8     4     5    12    11     6     9    15     1     2    10     3    14    13
     7     8     4    11    12     9     6    15     5     1     2    10     3    14    13
     7     8     4    11    12     9     6    15     5     1     2    10     3    14    13
     7     8     4    11    12     9     6    15     5    14     1     2    10     3    13
     7     8     4    11     9     6    15     5    14     1     2    10     3    13    12
```

Résumé

Ce travail concerne la résolution d'un problème d'ordonnancement de type Job Shop avec contrainte de transport dans lequel les produits sont transportés dans un sens unidirectionnel. Ce problème est un problème NP-Difficile, c'est pour cette raison qu'il n'existe pas de méthodes exactes permettant de résoudre de grandes instances de ce type de problème. Notre objectif dans ce travail est d'essayer d'adapter l'un des techniques des Métaheuristiques pour la résolution de ce type de problème toute en minimisant le temps total d'exécution Makespan. Un algorithme basé sur la méthode des Algorithmes Génétiques, est proposé permettant de donner des solutions plus proches de l'optimum. Cet algorithme proposé est par la suite évalué à travers un ensemble des exemples trouvés dans la littérature. La simulation informatique sous Matlab nous a permet de déduire la performance et l'efficacité de notre algorithme surtout pour les problèmes de petites tailles.

Mots clés : Ordonnancement – contrainte de transport - Job shop –Algorithmes génétiques métaheuristiques.

Abstract

This work concerns solving a type of scheduling problem with Job Shop transmission constraint in which the products are transported in a unidirectional sense. This problem is NP-hard problem is for this reason that there is no exact methods for solving large instances of this type of problem. Our goal in this work is to try to adapt one of Metaheuristics techniques for solving this problem all by minimizing the total time of execution Makespan. An algorithm based on Genetic Algorithms method is proposed to give more close to the optimum solutions. The proposed algorithm is then evaluated through a series of examples from the literature. The computer simulation in Matlab allows us to deduce the performance and efficiency of our algorithm especially for problems of small sizes.

Keywords: Scheduling - constrained transport - Job shop - genetic Algorithmes - metaheuristics.

ملخص

هذا العمل هو حل لمشكلة تنظيمية مع اجبارية تنقل المنتوج في اتجاه واحد هذا المشكل هو ذات درجة عالية من الصعوبة في هذا النوع من المشاكل لا توجد طرق تستطيع حل المشكل بدقة. هدفنا من هذا العمل حل المشكل بطرق تقريبية من اجل الهدف الاساسي وهو تقليص وقت الانتاج وقد اخترنا حل المشكل بطريقة الخوارزميات الجينية من اجل اعطائنا حلول تقريبية ولتقيم هذه الطريقة قمنا بدراسة عدة امثلة واستعملنا برنامج الكمبيوتر متلاب لمحاكات المشكل واعطائنا نتائج نقيم من خلالها فعالية هذه الطريقة.

كلمات مفتاحية المشاكل التنظيمية - الخوارزميات الجينية – الطرق التقريبية