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

FACULTE MATHEMATIQUES ET INFORMATIQUE
DEPARTEMENT INFORMATIQUE
N° :……..

DOMAINE : Mathématiques et Informatique
FILIERE : Informatique
OPTION : Informatique Décisionnelle et Optimisation

Mémoire présenté pour l'obtention
Du diplôme de Master Académique

Par: Zaiter Meriem

Intitulé

Insertion temps réel d'une production dans un atelier de type job shop

Soutenu devant le jury composé de :

Gasmi Abdelkader	Université de M'sila	Président
Dr. Mouhoub Nasser Eddine	Université de M'sila	Rapporteur
Guerna Abderrahim	Université de M'sila	Examinateur

Année universitaire : 2017 /2018

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

FACULTE MATHEMATIQUES ET INFORMATIQUE
DEPARTEMENT INFORMATIQUE
N° :……..

DOMAINE : Mathématiques et Informatique
FILIERE : Informatique
OPTION : Informatique Décisionnelle et Optimisation

Mémoire présenté pour l'obtention
Du diplôme de Master Académique

Par: Zaiter Meriem

Intitulé

Insertion temps réel d'une production dans un atelier de type job shop

Soutenu devant le jury composé de :

Gasmi Abdelkader	Université de M'sila	Président
Dr. Mouhoub Nasser Eddine	Université de M'sila	Rapporteur
Guerna Abderrahim	Université de M'sila	Examinateur

Année universitaire : 2017 /2018

Remerciements

Au début de ce modeste travail, je remercie le bon dieu de nous avoir orientés et guidés par sa merci et sa miséricorde pour réussir à élaborer ce travail.

Je tenais exprimer nos sincères remerciement à monsieur docteur Nacer eddine MOUHOUB qui nous à encadrer et aider par ces conseils et orientation.

Je suis honorée par la présence du jury d'avoir accepté d'examiner ce modeste travail et de porter leur critique objective et instructive remplie d'expérience.

Je tiens remercier tous les enseignants de département d'informatique.

Finalement, un vif remerciement à toutes personnes ayant participé de près ou de loin à l'élaboration de ce travail.

Dédicace

Avec les sentiments de gratitude les plus sincères

Je dédie ce modeste travail :

À mon père qu'il est vivent à mon cœur et jamais oublier pour son amour, sa patience, ses sacrifices, ses encouragements permanents pendant mes études.

À ma chère maman qui m'est la plus chère dans ce monde, qui est toujours éclairé mon chemin depuis le jour où j'ai vu la lumière par ses prières, son amour, ses sacrifices, ses encouragement par l'éducation qu'elle ma inculquée et tous les peines qu'elle s'est donnée pour moi pour ma réussite.

À ma grande sœur que je considérai comme une deuxième mère, elle mérite un remerciement chaleureux de m'avoir encouragé et soutenir jusqu'à la dernière minute.

Que dieu les recomposes, les gardes et les bénisses.

À mes adorables sœurs, à mes frères

À tout ma famille

À tous mes fidèles amis et mes collègues

« A great gratitude and respect from my heart to all the friends,

to all person who meet me with smile »

MERIAM.E

Table des matières

Table des matières …….. i

Liste des figures……... v

Liste des tableaux…… vi

Introduction générale …….. 1

CHAPITRE 1 Description du système de production …… 3

1.1Introduction ……... 3

1.2 Système de production ……. 3

1.2.1 Définition ……. 3

1.2.2Classification des systèmes de production…….. 3

1.2.2.1Classification selon la nature et le volume des flux physiques……... 3

1.2.2.2 Classification selon le mode de pilotage…… 5

1.2.3 Décomposition et organisation du système de production…… 5

1.2.4 La gestion de systèmes de production…….. 6

1.2.4.1Organisation hiérarchique de la gestion de production…….. 7

1.2.4.2 Fonction ordonnancement dans la gestion de production…….. 8

1.2.4.3 Flexibilité dans les systèmes de production……. 9

CHAPITRE 2 Problèmes d'ordonnancement dans les systèmes de production…… 11

2.1 Introduction …….. 11

2.2 Présentation générale des problèmes d'ordonnancement…….. 11

2.2.1 Définition de l'ordonnancement ……... 11

2.2.2 Les éléments d'un problème d'ordonnancement……... 12

2.2.2.1 Les tâches …… 12

2.2.2.2 Les ressources ……... 13

2.2.2.3 Les contraintes ……. 14

2.2.2.4 Les objectifs ou les critères d'évaluation …… 15

2.3 Notation des problèmes d'ordonnancement…….. 16

2.4 Typologie des ateliers…… 17

2.4.1 Les ateliers à une ressource…….. 17

2.4.2 Les ateliers à plusieurs ressources…… 18

2.4.2.1 Flow shop……. 18

2.4.2.2 Job shop……. 18

2.4.2.3 Open shop ……. 19

2.4.3 Les ateliers à ressources parallèles…… 19
2.5 Représentation des problèmes d'ordonnancement……... 20
2.5.1 Le diagramme de Gantt…….. 20
2.5.2 - Graphe Potentiel-Tâches……. 21
2.5.3 Méthode PERT……. 22
2.6 Complexité des problèmes d'ordonnancement……. 23
2.6.1 Complexité algorithmique …… 23
2.6.2 La complexité problématique …….. 23
2.7 Comment résoudre les problèmes d'ordonnancement?..... 24
2.7.1 Méthodes exactes …….. 24
2.7.2 Méthodes approchées …… 24
2.7.2.1 Les heuristiques…….. 24
2.7.2.2 Les Métaheuristiques……. 24
2.8 Problème d'ordonnancement Job Shop……. 27
2.8.1 Description du problème d'ordonnancement job shop…… 27
2.8.2 Définition formelle de problème d'ordonnancement job shop …… 27
2.8.2.1 Job shop classique …… 27
2.8.2.2 Problème de job shop flexible…….. 28
2.8.3Complexité du problème d'ordonnancement job shop…… 29
2.8.4 Méthodes de résolution du problème de Job Shop…… 30
2.8.4.1 Les différentes méthodes de résolution……. 30
2.8.4.2 Les Métaheuristiques……. 31
2.9 Ordonnancement d'atelier en temps réel…….. 31
2.9.1 Approches pour l'ordonnancement temps réel d'atelier……... 32
2.9.2 Méthode d'insertion de tâches……. 32
CHAPITRE 3 Les algorithmes génétiques pour la résolution de problèmes d'ordonnancement…… 33
3.1 Introduction …….. 33
3.2 Principe des Algorithmes Génétiques…… 33
3.2. 1 Historique……. 33
3.2.2 Présentation des algorithmes génétiques…… 33
3.2. 3 Description détaillée……. 35
3.2.3.1 Codage des données……... 35

3.2.3.2 Génération aléatoire de la population initiale…… 36
3.2.3.3 Evaluation : fitness……. 37
3.2.3.4 Sélection……. 37
3.2.3.5 Opérateur de croisement……... 38
3.2.3.6 Opérateur de mutation…… 39
3.3 Application des AG pour la résolution du problème d'ordonnancement job shop….. 40
3.3.1 Codage proposé……. 40
3.3.1.1 Codage direct ……. 40
3.3.1.2 Codage indirect …….. 41
3.3.2 Génération de la population initiale ……. 42
3.3.3 Le croisement pour les problèmes d'ordonnancement……. 42
3.3.4 La mutation pour les problèmes d'ordonnancement……. 43
CHAPITRE 4 Méthode d'insertion des nouvelles commandes…… 44
4.1 Introduction 44
4.2 Problème d'ordonnancement Job Shop (JSP)…… 44
4.2.1 Description du problème d'ordonnancement job shop…….. 44
4.3 Système à étudier et hypothèses proposés……. 44
4.3.1 Organisation du système et modèle de transport……. 44
4.3.1.1 Les machines…… 44
4.3.1.2 Les commandes ……. 45
4.3.1.3 Les tâches…….. 45
4.3.2 Formulation du problème et notations utilisée…….. 46
4.4 Méthode d'insertion des nouvelles commandes…… 47
4.4.1 Définition d'une position admissible …… 47
4.4.2 Procédure de décalage …….. 47
4.4.3 Création d'une position admissible et détermination du nouvel ordre ……. 48
4.4.4 Choix du critère et définition de la bonne solution ……... 48
4.4.5 Durée totale d'exécution Maksepan …… 49
4.5 Application de l'algorithme génétique dans l'insertion d'une tache dans un problème job shop…….. 49
4.5.1 Le codage basé sur les opérations……
4.5.2 Le croisement……... 50

4.5.3 Opérateur de Mutation…… 50

4.5.4 La sélection…….. 51

CHAPITRE 5 Conception et Réalisation…….. 52

5.1 Introduction …….. 52

5.2 Plateforme et outils de développement ……. 52

5.2.1 Environnement matériel …… 52

5.2.2 Environnement logiciel …… 52

5.2.2.1 Microsoft Visual Studio …… 52

5.2.2.2 C# …….. 53

5.3Interface graphique de l'application……... 53

Conclusion générale ……... 58

Listes de figures

Figure 1.1 systèmes de production……. 6

Figure 1.2 Organisation hiérarchique d'un système de gestion de production……. 8

Figure 1.3 les sous-fonctions de l'ordonnancement dans l'atelier…….. 9

Figure 2.1 Domaine concerné par l'ordonnancement…….. 12

Figure 2.2 Caractéristiques d'une tâche i……... 13

Figure 2.3 Typologie des problèmes d'ordonnancement par les ressources……. 14

Figure 2.4 Modèle machine unique ……. 17

Figure 2.5 Ordonnancement d'une séquence de jobs en Flow Shop……... 18

Figure 2.6 ordonnancement d'une séquence de jobs en Job Shop…….. 19

Figure 2.7 ordonnancement d'une séquence de jobs sur machines parallèle……. 20

Figure 2.8 Diagramme de Gantt……. 21

Figure 2.9 Graphe Potentiel-Tâches d'un ordonnancement…… 22

Figure 2.10 Classification des méthodes de résolution des problèmes d'ordonnancement…….. 26

Figure 2.11 Problème de job shop classique composé de 3 jobs et 5 machines…….. 28

Figure 2.12 Représentation d'un système de type Job-shop flexible à deux étages…… 29

Figure 3.1 Fonctionnement général de l'algorithme génétique……... 35

Figure 3.2 Codage binaire d'un chromosome……. 36

Figure 3.3 Codage réel d'un chromosome…….. 36

Figure 3.4 Codage par valeurs d'un chromosome……... 36

Figure 3.5 La roulette de sélection……. 37

Figure 3.6 Croisement avec un point de deux chromosomes…… 38

Figure 3.7 Croisement avec deux points de deux chromosomes……. 38

Figure 3.8 Croisement avec uniforme……. 39

Figure 3.9 Une mutation…… 39

Figure 4.1 Diagramme de Gant d'un ordonnancement prévisionnel…… 47

Figure 4.2 Exemple d'une opération de décalage effectué pour insérer la tâche O_i^a… 49

Figure 5.1 l'interface de Visual Studio 2013 …… 53

Figure 5.2 Interface graphique…… 54

Figure 5.3 Format général des instances……. 54

Figure 5.4 Interface graphique de l'exemple …… 55

Figure 5.5 Interface après exécution du plan prévisionnel …… 56

Figure 5.6 Interface pour ajouter nouvelle commande ……. 57

Liste des tableaux

Tableau 2.1 Gamme opératoire des jobs …… 19

Tableau 2.2 La table initiale d'ordonnancement T pour la réalisation d'un graphe potentiel-tâches…….. 21

Introduction générale

Dernièrement, les tendances actuelles dans le domaine industriel sont orientées vers l'amélioration des performances des systèmes manufacturiers en termes d'adaptation rapide vu les fluctuations du marché et les perturbations internes. Une telle vision est étroitement sollicitée par la mondialisation exacerbant une concurrence acharnée désormais internationale.

En effet, la production devient de plus en plus diversifiée et les entreprises cherchent régulièrement à accroître leurs parts de marché ainsi que les marges de profit. Sous ces conditions, la fonction ordonnancement est considérée incontestablement dans la hiérarchie décisionnelle de l'entreprise comme un pilier solide pour prendre les décisions les plus appropriées tant pour les investisseurs et producteurs que pour les consommateurs.

L'ordonnancement est une branche de la recherche opérationnelle et de la gestion de la production, joue un rôle essentiel au sein de l'entreprise, permettant à cette dernière de satisfaire les exigences et les contraintes imposées par ses clients en termes de qualité, 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 jusqu'à l'informatique, Les modèles d'ordonnancement diffèrent selon les technologies utilisées et les contraintes auxquelles est soumis le système. Les problèmes d'ordonnancement les plus répandus dans la littérature sont les problèmes d'ordonnancement d'atelier de type job shop. Ce problème consiste à trouver une séquence optimale pour l'exécution de n jobs sur m machines afin d'optimiser une fonction objectif. La plupart des problèmes d'ordonnancement sont classés NP-difficiles.

Dans ce travail, nous nous intéressons plus particulièrement aux problèmes d'ordonnancement en temps réel pour un system de type Job Shop . Ce système est soumis à une perturbation de l'environnement représentée par l'occurrence des nouvelles commandes urgentes, qu'il doit les exécuter.

Le problème imposé ici est comment le système doit il réagir lors de l'apparition aléatoire d'une nouvelle commande ?

Les méthodes de résolution des problèmes d'ordonnancement puisent dans toutes les techniques de l'optimisation combinatoire, dans ce mémoire nous nous intéressons à l'adaptation des algorithmes génétique pour résoudre le problème imposé.

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

Le premier chapitre est réservé à la présentation les notions de base concernant le système de production et la gestion de production, en mettant l'accent sur le rôle de l'ordonnancement au sein de ces systèmes.

Le deuxième chapitre est réservé au cadrage de la problématique générale de l'ordonnancement par la présentation des points essentiels : définitions, typologies, modélisations, critères d'optimisation et organisation des ateliers.

Le troisième chapitre est consacré les algorithmes génétiques de base, leurs principes de fonctionnement et leurs particularités.

Dans le quatrième chapitre, nous commençons par le problème d'ordonnancement job shop, suivi d'une description du problème à étudier.

Le cinquième et dernier chapitre portera sur l'implémentation de notre application ainsi que l'élaboration des tests expérimentaux. Nous finirons notre travail par une conclusion générale.

CHAPITRE 1

Description du système de production

1.1 Introduction

Nous allons présenter dans ce chapitre les notions fondamentales concernant les systèmes de production et leurs classifications et particulièrement les systèmes flexibles de production Pour cela, nous rappelons la définition de flexibilité, nous étudions ensuite les Dimensions de flexibilité.

1.2 Système de production

1.2.1 Définition

Le système de production est composé d'un ensemble de ressources organisées qui interagissent et interfèrent dans le but de produire des biens ou des services. [1] [2]

D'une autre manière, un système de production est un ensemble de ressources réalisant une activité de production. La production représente la transformation qui s'effectue par une succession d'opérations utilisant des ressources (machines et opérateurs) et modifiant les matières premières ou composants entrant dans le système de production afin de créer les produits finis sortant de ce système et destinés à être consommés par des clients. Les modifications peuvent porter sur la forme du produit, sa structure, son apparence, etc. Les ressources appartenant au système de production mobilisées pour réaliser l'activité de production peuvent être des machines, des opérateurs, de l'énergie, des informations, des outillages… [3].

1.2.2 Classification des systèmes de production

On peut classer les systèmes de production en deux types, La première classification repose sur la nature et le volume des flux physiques dans le système, et l'autre selon le mode de pilotage. [4]

1.2.2.1 Classification selon la nature et le volume des flux physiques

La première classification est composée sur la nature du système physique et le volume des produits fabriqués par ce dernier. Dans ce cadre, on distingue principalement trois types de systèmes :

Systèmes à flux continu : dans ces systèmes, la caractéristique principale est la circulation des matières en flux continu. C'est-à-dire 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 [5].

Systèmes à flux discret: dans ces systèmes, les produits peuvent être distingués individuellement (production discrète). De plus, des zones de stockage temporaire entre deux postes de travail sont utilisées. C'est le cas des entreprises manufacturières. C'est cette catégorie de systèmes qui nous intéresse dans notre étude où trois classes peuvent être distinguées [4]:

- Les systèmes de production en grande série ou de masse : cette classe se caractérise par le lancement en production, dans les mêmes délais, de grands volumes de produits similaires. L'ordre de passage des produits sur les ressources (machines, stocks, etc.) étant toujours le même, celles-ci peuvent être placées dans un ordre figé dépendant du produit à fabriquer.
- 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 ne permet pas une spécialisation des moyens de production. Les différents produits suivent leur propre chemin sur des ressources communes de flexibilité élevée souvent regroupées par fonctionnalités équivalentes.
- Les systèmes de production unitaire : pour ce type de systèmes, la taille du produit ou la demande impose une production de très faible quantité. Dans ce cas, la tâche principale de la production consiste à réunir les moyens nécessaires au bon moment et au bon endroit. [4]

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. [5]

1.2.2.2 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 : La deuxième classification des systèmes de production est étroitement liée à la stratégie de pilotage utilisée. En effet, en se situant au niveau opérationnel et sur la base du mode de déclenchement de la production, cette classification sépare les systèmes fonctionnant à flux tirés de ceux fonctionnant à flux poussés.

Les systèmes à flux tirés : dans ce contexte, la production est déclenchée par la consommation des produits finis. D'une manière générale, on peut distinguer deux types de fonctionnement. Le premier consiste à maintenir un stock minimum spécifié de produits finis. Dans ce cas, on parle d'une production sur stock.

Autrement dit, si un produit quitte le stock, un ordre de fabrication est lancé au système afin de pouvoir le reconstituer. Dans le deuxième type, la production est déclenchée par la réception d'une commande. En d'autres termes, ce type de fonctionnement vise à maintenir un stock de produits finis nul. [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]

1.2.3 Décomposition et organisation du système de production

Les systèmes de production industrielle peuvent être des systèmes très complexes et difficiles à gérer au vu de toutes leurs composantes fonctionnelles (fabrication, achat, distribution, maintenance...) [3]. En effet, ils peuvent se décomposer en trois sous-systèmes : le sous-système physique de production, le sous-système d'information et le sous-système de décision Figure 1.1, qui s'intègrent en vue d'assurer la pérennité et la compétitivité de l'entreprise. [2]

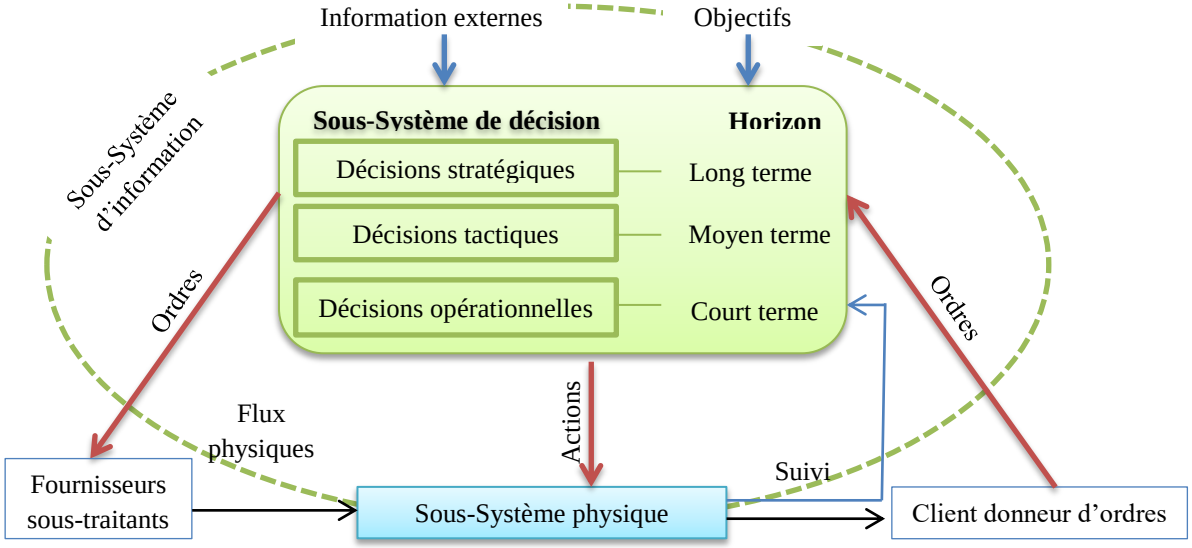


Figure 1.1 systèmes de production [2]

Le sous-système physique de production : englobe toutes les différentes ressources nécessaires (ressources humaines et physiques) pour la transformation des matières premières en produits finis. [2]

Le sous-système de décision : Son objectif est contrôle le système physique de production à travers l'organisation des différentes activités en prenant des décisions basées sur les données transmises par le système informationnel. [2][5]

Ce sous-système est aussi divisé en trois niveaux:

- Niveau stratégique : qui concerne les décisions à long terme.
- Niveau tactique : où les décisions sont prises au moyen terme.
- Niveau opérationnel : limité sur le court terme.

Le sous-système d'information : joue le rôle d'interface entre le sous-système de décision et le sous-système physique de production d'une part. D'autre part, il intervient à l'intérieur du sous-système de décision, pour la gestion des informations utilisées lors de la prise de décision. Son rôle est de collecter, stocker et transmettre des informations de différents types. [2]

1.2.4 La gestion de systèmes 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 [3]. Comme présenté précédemment, le système de gestion de production est une fonction complexe, constitué par la partie du sous-système de décision et du sous-système

d'information, ils traitent les fonctions rattachées directement à la production à savoir, les achats, les approvisionnements, la planification, la gestion des ressources, la maintenance, etc.

La commande du système de gestion au sous-système physique est caractérisée par la transformation des informations à caractère commercial en ordres de fabrication et ordres d'approvisionnements. Le système de gestion de production reçoit un feedback sous forme d'information de suivis du sous-système physique, qui sert effectivement à piloter ce dernier. [2]

1.2.4.1Organisation hiérarchique de la gestion de production

La gestion de la production s'effectue par un ensemble de décisions qui peuvent être hiérarchisées suivant un ordre temporel, dont plusieurs activités peuvent être attribuées au système de gestion. En général, trois niveaux hiérarchiques de la gestion de production sont couramment retenus, à savoir les niveaux ; stratégique, tactique et opérationnel [2] [3] :

- **Le niveau opérationnel :** C'est le niveau d'exécution du système qui sert à une gestion quotidienne pour faire face à la demande du jour le jour, dans le respect des décisions tactiques. Il génère des décisions à court et à très court terme qui déterminent l'ordonnancement et le pilotage.
- **Le niveau tactique :** Il s'agit de décisions à moyen terme qui s'effectuent sur plusieurs mois. Elles assurent la liaison entre le niveau stratégique et le niveau opérationnel. L'objectif est de produire au moindre cout pour satisfaire la demande prévisible, en s'inscrivant dans le cadre fixe par le plan stratégique de l'entreprise.
- **Le niveau stratégique :** Il s'agit de la formulation de la politique et la stratégie de l'entreprise à long terme, qui porte essentiellement sur la gestion des ressources durables, afin que celles-ci soient en mesure d'assurer la pérennité de l'entreprise sur plusieurs années.

La Figure 1.2 présente l'interaction qui existe entre les activités du système de gestion et les niveaux de planification de ces activités.

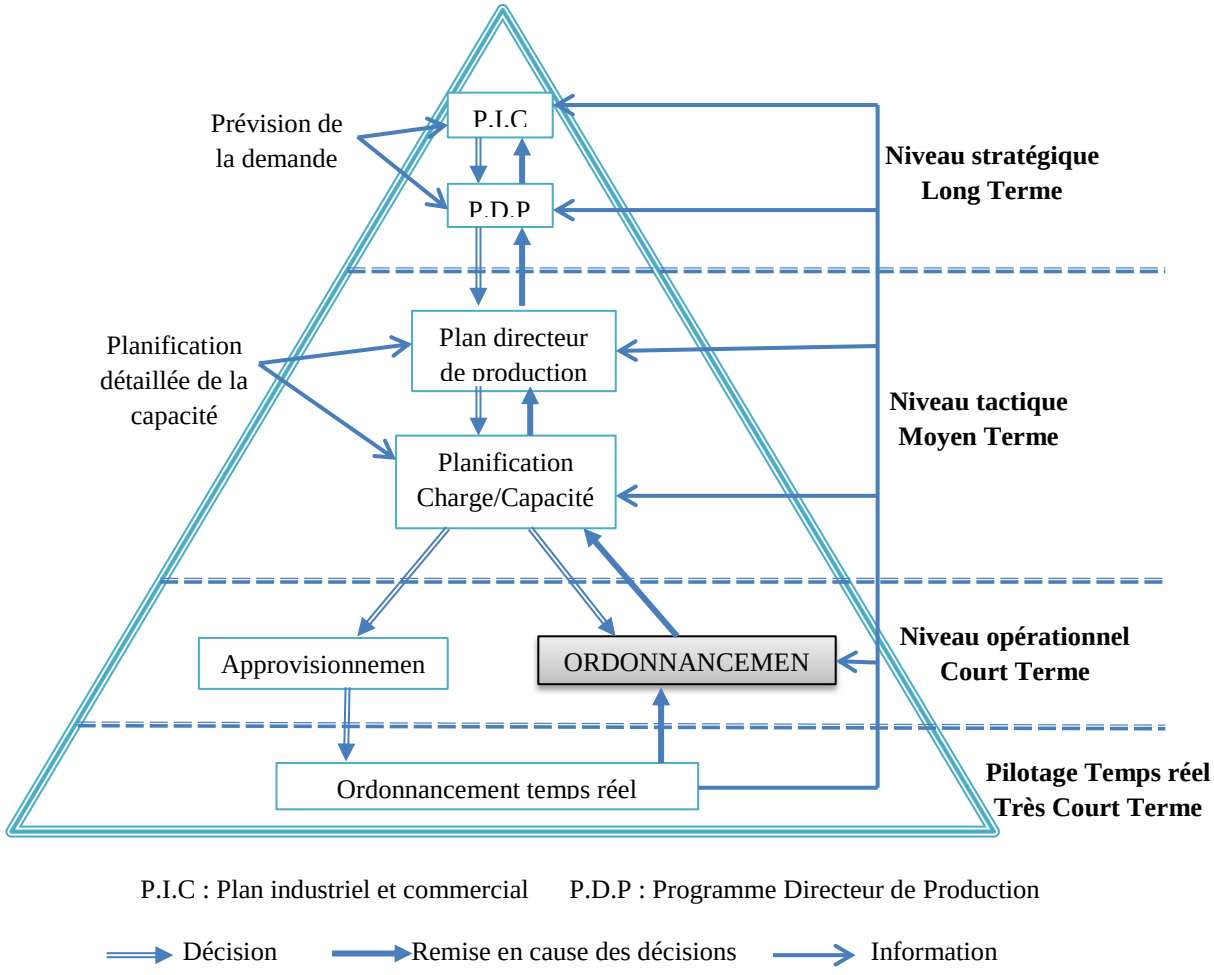


Figure 1.2 Organisation hiérarchique d'un système de gestion de production [2]

1.2.4.2 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 :

• 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).

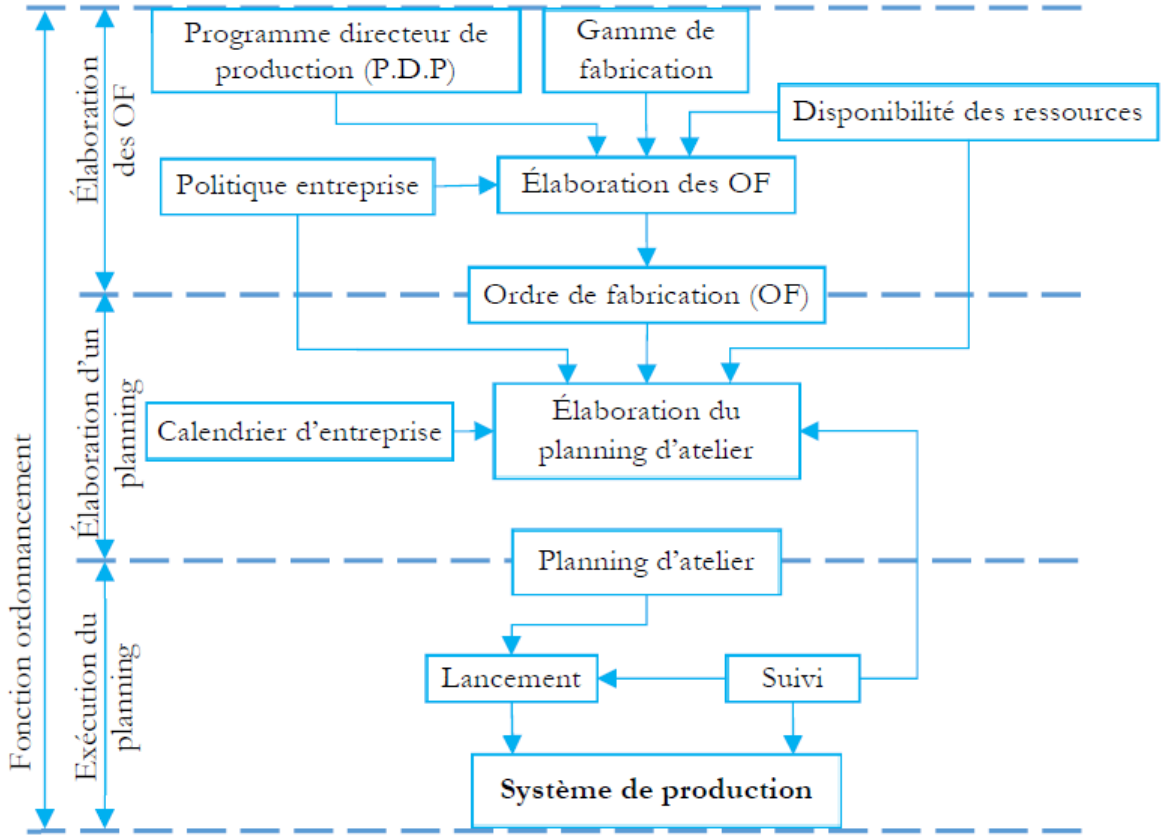


Figure 1.3 les sous-fonctions de l'ordonnancement dans l'atelier [2]

1.2.4.3 Flexibilité dans les systèmes de production

La flexibilité est la notion principale utilisée dans la conception des systèmes automatisés modernes de fabrication comme les systèmes flexibles de production. [6]

Nagarur a défini la flexibilité des systèmes de production par la capacité d'un système à s'adapter rapidement à des changements de facteurs clefs tels que le produit, le process, la charge et les pannes des machines. [5]

On peut définir la flexibilité par l'ensemble des propriétés et qualités d'un système manufacturier qui peut supporter des changements dans les types de produits et la capacité de production. Ces changements peuvent être :

- Internes : comme les pannes des équipements, pannes des systèmes informatiques (logiciels de gestion), absentéisme des travailleurs, variations des temps de fabrication…etc.
- Externes : comme le changement dans la conception des produits, la complexité de conception des produits, la variation de la demande. [6]

Pour le domaine de l'ordonnancement, indiquent que la flexibilité peut être exprimée comme l'existence de modifications possibles dans un ordonnancement, calculé hors-ligne, entraînant une perte de performance acceptable. Elle peut aussi être associée à un voisinage de solutions pouvant être examiné lors de l'exécution, ou à une famille d'ordonnancements, sans privilégier un ordonnancement en particulier.

CHAPITRE 2

Problèmes d'ordonnancement dans les systèmes de production

2.1 Introduction

Le but de ce chapitre, introduire quelques notion sur les problèmes d'ordonnancement dans les systèmes de production, Nous rappelons d'abord les différents paramètres et classifications dans tel problème, puis nous expliquons sur la théorie de la complexité, ensuite les principes généraux des méthodes de résolution approchée.

2.2 Présentation générale des problèmes d'ordonnancement

2.2.1 Définition de l'ordonnancement

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. [7]

Selon [8], « Ordonnancer c'est programmer l'exécution d'une réalisation en attribuant des ressources aux tâches et en fixant leurs dates d'exécution. ». Présentons une autre définition qui est plus explicite : « Un ordonnancement constitue une solution au problème d'ordonnancement. Il décrit l'exécution des tâches et l'allocation des ressources au cours du temps, et vise à satisfaire un ou plusieurs objectifs. Plus précisément, on parle de problème d'ordonnancement lorsqu'on doit déterminer les dates de début et les dates de fin des tâches, alors qu'on réserve le terme de problème de séquencement au cas où l'on cherche seulement à fixer un ordre relatif entre les tâches qui peuvent être en conflit pour l'utilisation des ressources. Un ordonnancement induit nécessairement un ensemble unique de relations de séquencement. »

L'ordonnancement se déroule en trois étapes qui sont:

- la planification qui vise à déterminer les différentes opérations à réaliser, les dates correspondantes et les moyens matériels et humains à y affecter.
- l'exécution qui consiste à mettre en œuvre les différentes opérations définies dans la phase de planification.

- le contrôle qui consiste à effectuer une comparaison entre la planification et l'exécution, soit au niveau des coûts, soit au niveau des dates de réalisation. [9]

Les problèmes d'ordonnancement existent dans de nombreux secteurs d'activités comme la gestion de production, dans l'industrie manufacturière on encore les systèmes informatiques, où les tâches représentent les programmes et les ressources sont les processeurs ou la mémoire, la conception des emplois des temps, la figure 2.1 montre les différents domaines concerné par l'ordonnancement.

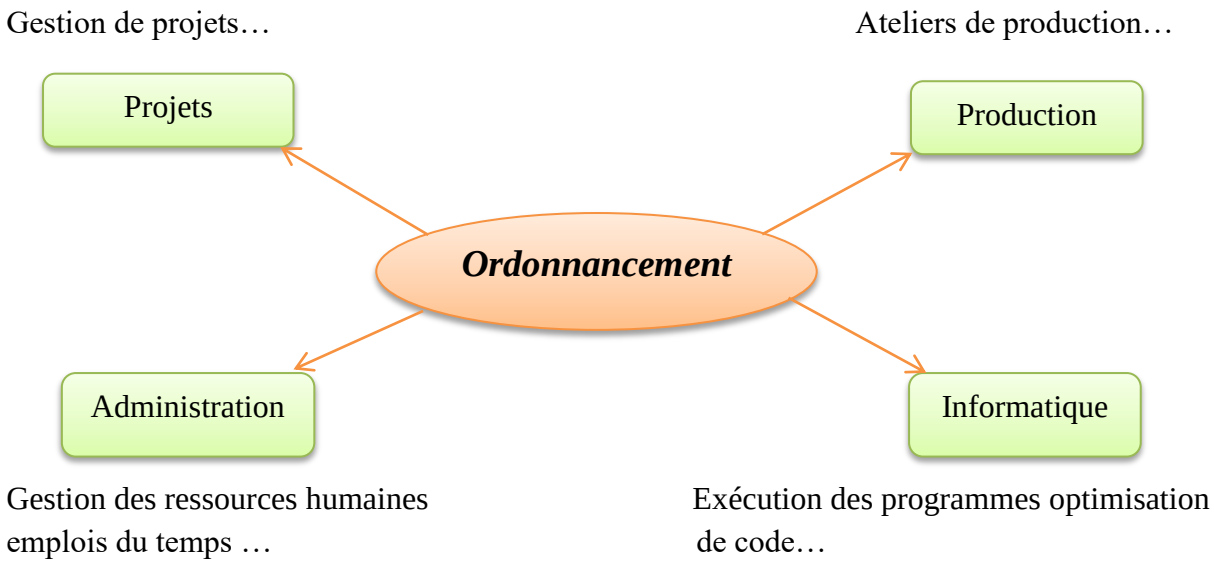


Figure 2.1 Domaine concerné par l'ordonnancement

2.2.2 Les éléments d'un problème d'ordonnancement

Les principaux éléments d'un problème d'ordonnancement sont les tâches, les ressources, les contraintes et les critères. Résoudre un problème d'ordonnancement consiste à programmer l'exécution des tâches et à allouer les ressources requises afin d'optimiser un ou plusieurs critères, tout en respectant un ensemble de contraintes. [9]

2.2.2.1 Les tâches (les opérations)

La fabrication de produits dans un atelier de production nécessite l'exécution d'un ensemble d'opérations élémentaires ou tâches. Une tâche i est localisée dans le temps par une

date de début d'exécution ti et/ou par une date de fin ci et une durée d'exécution pi telle que $p_i=c_i-t_i$ [9], et qui utilise des ressources k avec une intensité a_i^k. [10]

Généralement trois paramètres caractérisent une tâche « l'opération :

- La durée opératoire p_i (processing time) :c'est la durée d'exécutions de la tâche.
- La date de disponibilité r_i (release time) : c'est la date de début au plus tôt.
- La date d'échéance d_i (due date) : c'est la date de fin au plus tard.

La figure 2.2 illustre les caractéristiques temporelles d'une tâche.

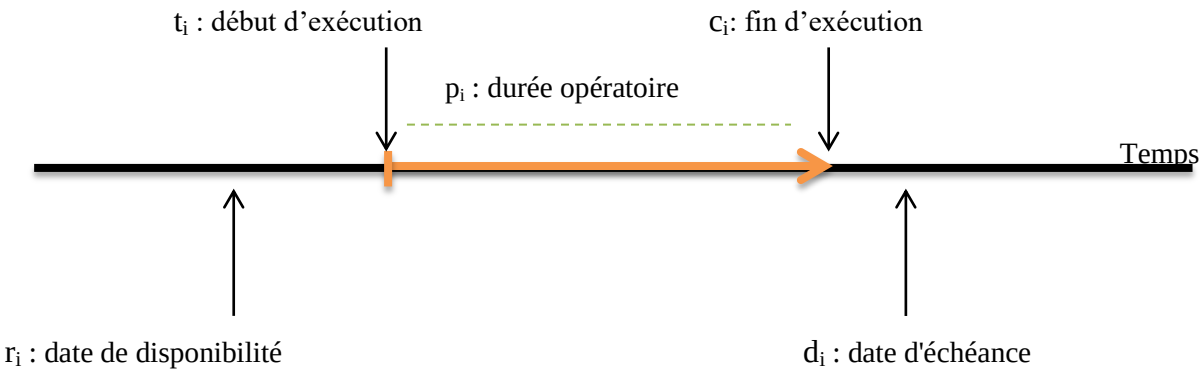


Figure 2.2 Caractéristiques d'une tâche i [10]

On peut classer les tâches comme suit :

- les tâches morcelables (préemptives) : qui peuvent se réaliser par morceaux et peuvent être exécutées en plusieurs fois, facilitant ainsi la résolution de certains problèmes.
- les tâches non morcelables (indivisibles) : qui doivent être exécutées en une seule fois et ne sont interrompues qu'elles soient complètement terminées. [7]

2.2.2.2 Les ressources

Pour l'exécution des tâches, ces dernières requièrent certaines ressources telles que des machines, la main d'œuvre, les moyens financiers, etc. [8]

Une ressource k est définie comme un moyen technique ou humain requis pour la réalisation d'une tâche. [9]

Dans un atelier, on trouve deux types de ressources : [8]

- Une ressource est consommable si, après avoir été allouée à une tâche, elle n'est plus disponible pour les tâches suivantes. Le cas pour l'argent, la matière première, etc.
- Une ressource est renouvelable si, après avoir été allouée à une tâche, elle redevient disponible après la fin de cette tâche pour les tâches suivantes. C'est le cas pour les

machines, les processeurs, les fichiers, le personnel, etc [8]. La quantité de ressources utilisables à chaque instant étant limitée. [9]

- les ressources disjonctives (ou non partageables) allouées à une tâche à la fois (machine outils, robot manipulateur).

- les ressources cumulatives (ou partageables) peuvant être utilisées par plusieurs tâches simultanément (équipe d'ouvriers, poste de travail). [9]

La figure 2.3 [8] illustre la nature des ressources prises en considération permet de dresser une typologie des problèmes d'ordonnancement.

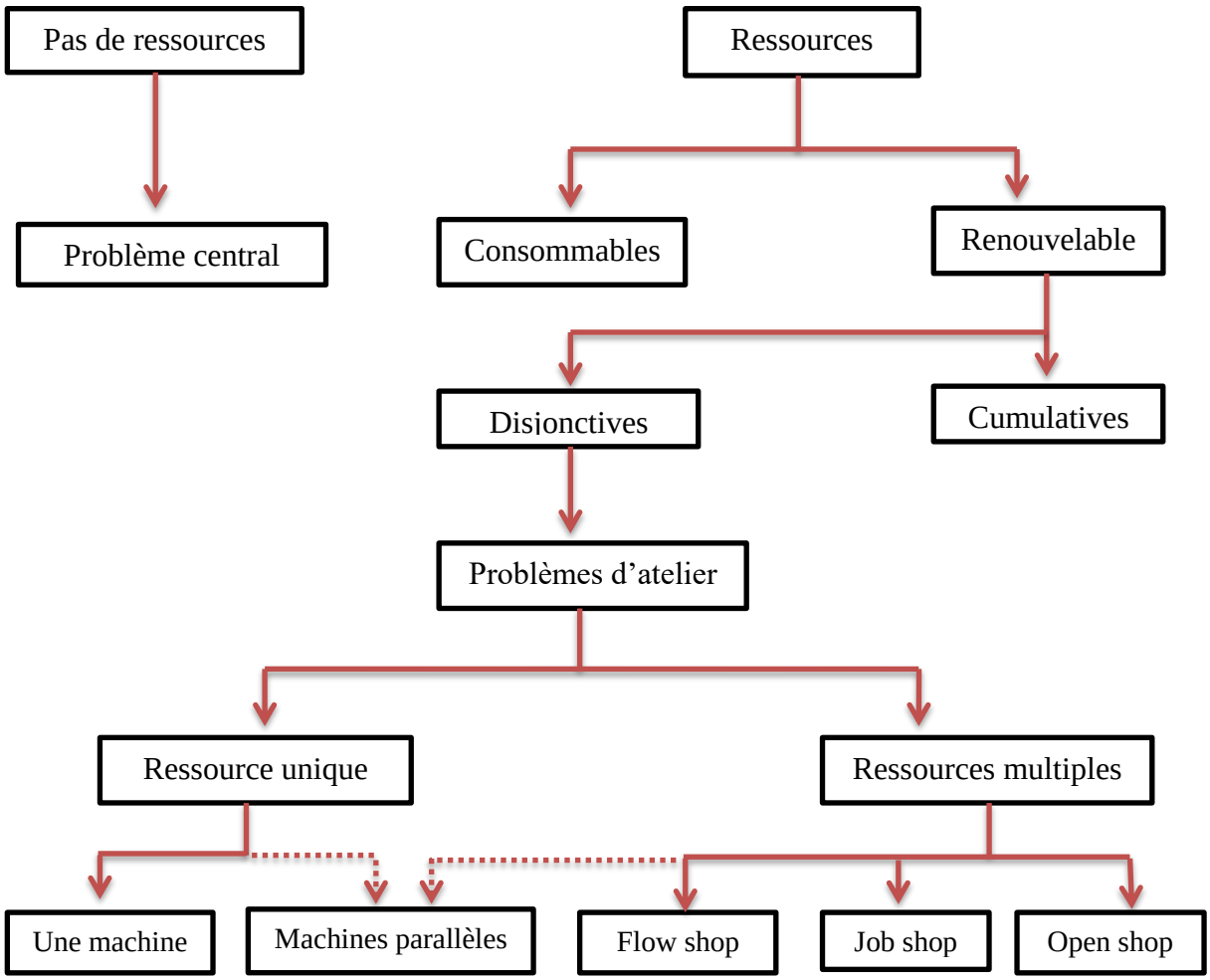


Figure 2.3 Typologie des problèmes d'ordonnancement par les ressources [8]

2.2.2.3 Les contraintes

Une contrainte est une restriction sur les valeurs que peuvent prendre une ou plusieurs variables de décision sur le temps (variable d'ordonnancement) ou bien sur les ressources (variables d'affectation). [8]

On peut distinguer quatre grandes catégories de contraintes : temporelles, technologiques, d'enchaînement et de ressources. [11]

Contraintes temporelles : Le premier type concerne les délais de fabrication imposés. Ces contraintes sont issues des impératifs de gestion et relatives aux dates limites des tâches ou du projet entier. On a surtout :

· La date de disponibilité r_i (avant laquelle la tâche ne peut pas commencer).

· La date d'échéance d_i (avant laquelle la tâche doit être achevée).

Contraintes technologiques : Le deuxième type correspond aux contraintes technologiques, en général décrites dans les gammes de fabrication des produits.

Contraintes d'enchaînement : Nous qualifions de contrainte d'enchaînement ou de succession, une contrainte qui lie le début ou la fin de deux activités par une relation linéaire.

Ce sont des contraintes imposées généralement par la cohérence technologique (les gammes opératoires dans le cas d'ateliers) qui décrivent des positionnements relatifs devant être respectés entre les tâches. [11]

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

Les contraintes relatives aux ressources peuvent être disjonctives, induisant une contrainte de réalisation des tâches sur des intervalles temporels disjoints pour une même ressource, ou cumulatives impliquant la limitation du nombre de tâches à réaliser en parallèle. [7]

2.2.2.4 Les objectifs ou les critères d'évaluation (La fonction économique)

Un critère correspond à des exigences qualitatives et quantitatives à satisfaire permettant d'évaluer la qualité de l'ordonnancement établi [9]. Les critères que doit satisfaire un ordonnancement sont variés. D'une manière générale, on distingue plusieurs classes d'objectifs concernant un ordonnancement.

-Les objectifs liés au temps : On trouve par exemple la minimisation du temps total d'exécution, du temps moyen d'achèvement, des durées totales de réglage ou des retards par rapport aux dates de livraison.

- Les objectifs liés aux ressources : maximiser la charge d'une ressource ou minimiser le nombre de ressources nécessaires pour réaliser un ensemble de tâches sont des objectifs de ce type.

- Les objectifs liés au coût : ces objectifs sont généralement de minimiser les coûts de lancement, de production, de stockage, de transport, etc. [11]

On peut classer les critères en critères réguliers et en critères irréguliers [9].

- Les critères réguliers : un critère est dit régulier s'il est une fonction décroissante des dates de fin d'exécution des opérations. Par exemple, la durée globale de fabrication d'un produit [8], minimisation du maximum des retards sur les dates d'achèvement, minimisation de la moyenne des retards sur les dates d'achèvement. [2]
- Les critères irréguliers : Ils ne sont pas des fonctions monotones des dates de fin d'exécution des opérations. Parmi ces types de critères, on cite:

-La minimisation des encours ;

-La minimisation du coût du stockage des matières premières ;

- délai de fabrication, avances, retard,
- L'équilibrage des charges des machines. [7].

2.3 Notation des problèmes d'ordonnancement

Plusieurs notations ont été introduites dans la littérature pour spécifier un problème d'ordonnancement. Un schéma de classification propose par Graham et al. (1979), puis repris par Blazewicz et al. (1996). Il est rapidement imposée comme faisant référence. Il permet en effet de caractériser un problème d'ordonnancement de manière précise. Cette notation structure les données d'un problème d'ordonnancement sur la base d'une notation à trois champs distincts α, β et γ (α|β|γ) [2] :

- le champ α, permettant de décrire le type d'atelier, le nombre de jobs à réaliser et le nombre de machines disponibles,
- le champ β, caractérisant les conditions d'exécution des jobs ainsi que les états des ressources présentes dans l'atelier. Il indique en particulier la présence ou non des contraintes de précédence entre les tâches et la possibilité de tolérer la préemption,
- le champ γ, dédié au(x) critère(s) à optimiser.

Pour l'exemple suivant : J, 10, 6|*Prec*, r_i |C_{max}

Il s'agit d'un problème d'ordonnancement d'un atelier de type job-shop, ayant dix jobs et six machines disponibles. Le deuxième champ indique que les jobs présentent une contrainte de précédence, *Prec*, et une contrainte r_i de dates de début au plus tôt. En plus, la préemption est interdite (puisqu'elle n'est pas mentionnée dans le champ *prem*). Le dernier champ montre que l'objectif est de minimiser le makespan, $C_{max.}$ [9].

2.4 Typologie des ateliers

Une classification des problèmes d'ordonnancement dans un atelier peut s'opérer 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. [7].

On distingue trois catégories. La première regroupe les problèmes pour lesquels chaque tâche nécessite une seule machine (ressources). Dans la deuxième, chaque tâche demande plusieurs machines pour son exécution. Dans la troisième catégorie, les machines sont regroupées en étages distincts. [9]

2.4.1 Les ateliers à une ressource

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

Parmi les problèmes rencontrés dans un environnement à une machine, on peut citer les problèmes de minimisation du retard maximum, du nombre de tâches en retard, ou de la somme des retards. La Figure (2.4) illustre le modèle machine unique.

Malgré leur apparente simplicité, les problèmes à une machine sont NP-difficiles au sens fort. [2]

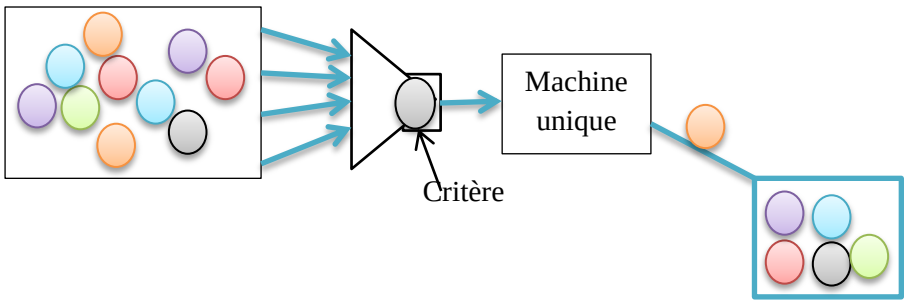


Figure 2.4 Modèle machine unique [2]

2.4.2 Les ateliers à plusieurs ressources

Ces ateliers sont composés d'un ensemble de *m* machines (ou ressources) dont chacune ne peut réaliser, à elle seule, l'ensemble des opérations. Suivant le mode de passage sur les différentes machines, trois types d'ateliers sont distingués [9]. Les problèmes d'ordonnancement sont généralement classes en trois principaux modèles dépendamment du nombre d'opérations que requièrent les jobs : des modèles à une opération (machine unique et machines parallèles) et des modèles a plusieurs opérations (flow shop, open shop et job shop) [2].

2.4.2.1 Flow shop

Dans un tel atelier, appelé aussi atelier en ligne ou à cheminement unique, toutes les opérations passent par toutes les machines dans le même et unique ordre. La figure 2.5 illustre un atelier de type flow shop. Dans les ateliers de type flow-shop hybride, une machine peut exister en plusieurs exemplaires identiques fonctionnant en parallèle.

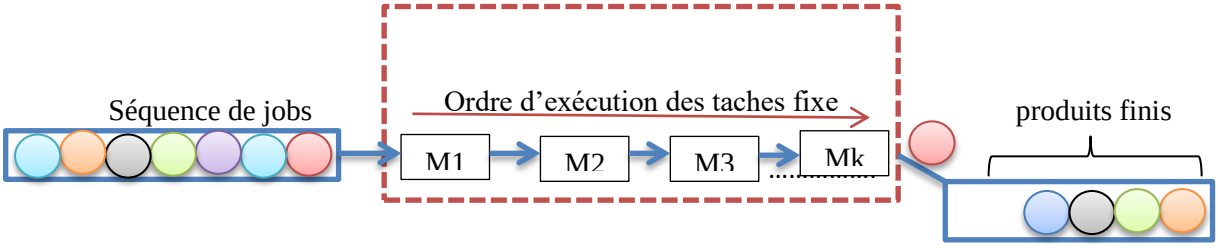


Figure 2.5 ordonnancement d'une séquence de jobs en Flow Shop [2]

2.4.2.2 Job shop

Appelés également ateliers a cheminement multiple, ce sont des ateliers ou les opérations sont réalisées selon un ordre défini par la gamme de fabrication prédétermine, variant selon le job à exécuter, Tableau (2.1) et Figure (2.6). Les premiers résultats théoriques du job shop, concernant la résolution d'un problème a deux machines et plus tard a trois machine, revient aux années cinquante. La première définition formelle de cet atelier a paru dans l'ouvrage de Muth et Thompson 1963. Certains problèmes job shop, proposes a cette époque, ont été résolus dans les années soixante-dix. En revanche, il a fallu attendre les années quatre-vingt pour d'autres. Jusqu'à ce jour cet atelier reste le contre d'intérêt d'un nombre important de chercheurs.

Le job-shop flexible est une extension du modèle job-shop classique, sa particularité réside dans le fait que plusieurs machines sont potentiellement capables de réaliser un sous-ensemble d'opérations.

	1	2	3	4	1	2	3	4	1	2	3	4
J_1	M_1	M_2	M_3		10	8	4		O_{11}	O_{21}	O_{31}	
J_2	M_2	M_1	M_4	M_3	8	3	5	6	O_{22}	O_{12}	O_{42}	O_{32}
J_3	M_1	M_2	M_4		4	7	3		O_{13}	O_{23}	O_{43}	
	Machines				Durée opératoire (p_{ij})				Opérations			

Tableau 2.1 : Gamme opératoire des jobs [2]

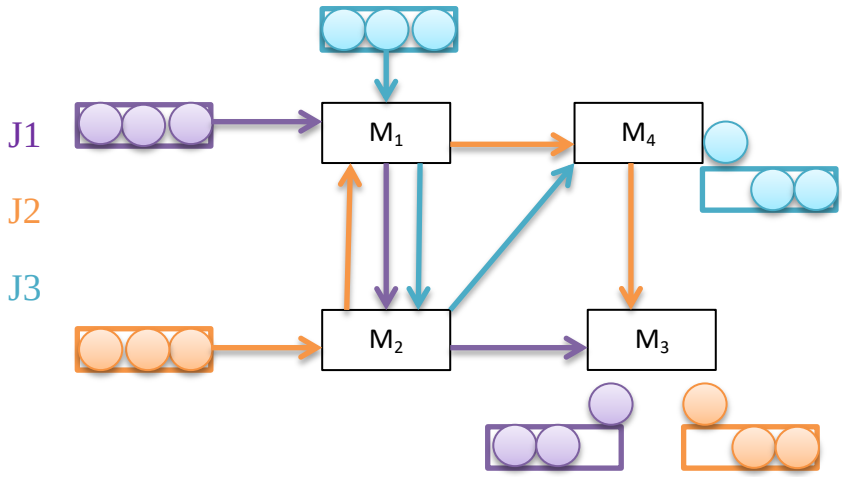


Figure 2.6 ordonnancement d'une séquence de jobs en Job Shop [2]

2.4.2.3 Open shop

C'est un atelier à cheminement libre, l'ordre des opérations n'étant pas fixé a priori; le problème d'ordonnancement consiste, d'une part, à déterminer le cheminement de chaque produit et, d'autre part, à ordonnancer les produits en tenant compte des gammes trouvées, ces deux problèmes pouvant être résolus simultanément. Ce type d'ateliers n'est pas couramment utilisé dans les entreprises.

2.4.3 Les ateliers à ressources parallèles

Dans ces ateliers, les machines sont regroupées en étages distincts. Chaque machine appartient à un seul étage et toutes les machines d'un même étage peuvent effectuer la même opération. Ce type d'atelier peut être divisé en trois sous-catégories selon la vitesse d'exécution des machines:

- les ateliers à machines identiques : toute tâche peut s'exécuter sur n'importe quelle machine avec une même durée opératoire,

- les ateliers à machines uniformes : chaque machine possède sa propre vitesse, indépendamment de la durée de la tâche à exécuter,
- les ateliers à machines indépendantes : la vitesse de chaque machine dépend de l'opération à effectuer.

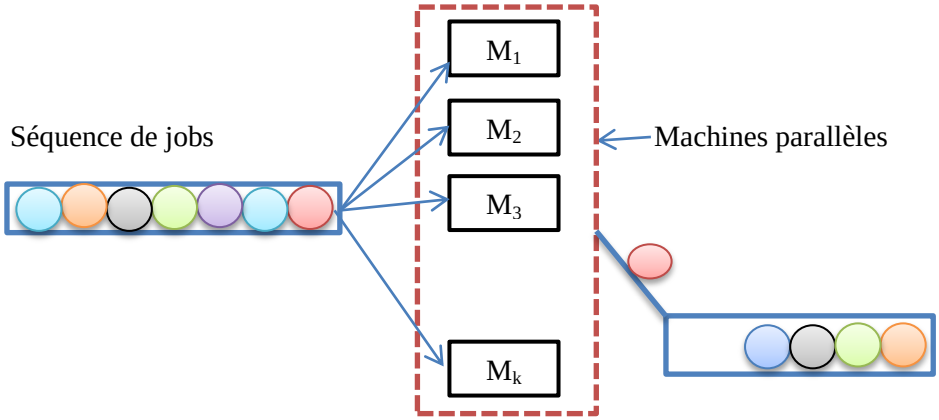


Figure 2.7 ordonnancement d'une séquence de jobs sur machines parallèle [2]

2.5 Représentation des problèmes d'ordonnancement

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

2.5.1 Le diagramme de Gantt

Le diagramme de Gantt est un outil permettant de représenter les séquences de traitement des jobs sur chaque machine requise, en précisant la date de début et de fin de chacune d'elles. Il s'agit d'un outil élabore en 1917 par Henry L. Gantt. En effet, le diagramme de Gantt représente en ordonnée la liste des taches, notées O_i à exécuter par les machines notées M_i et en abscisse l'échelle du temps par lequel on précise la date de début, la durée de la tâche et la date de fin de chaque tache. A titre d'illustration une solution réalisable (avec critère Makespan) du problème Tableau 2.1 est donnée par la Figure 2.8.

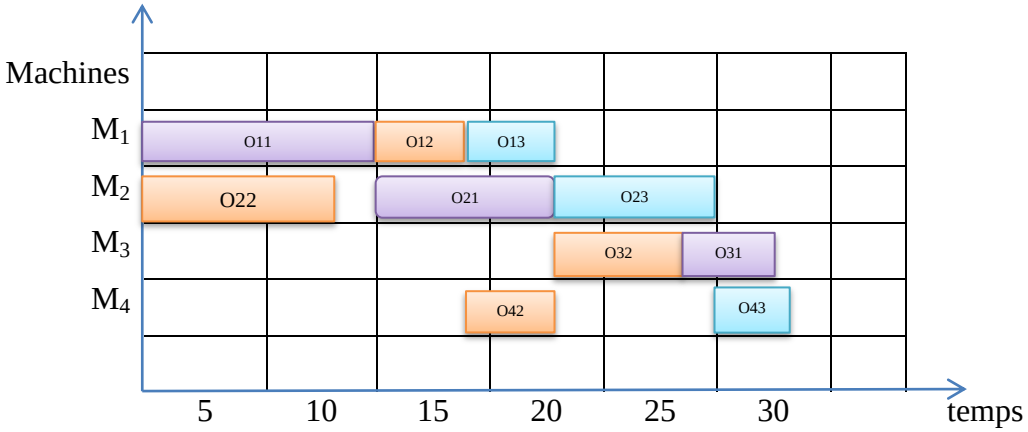


Figure 2.8 Diagramme de Gantt [2]

2.5.2 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 [7].

Dans ce genre de modélisation, les tâches sont représentées par des nœuds et les contraintes par des arcs, comme le montre la figure 1.3.

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.

Tâches	Durées	Antériorités
A	0	-
A	6 mois	A
B	3 mois	A
C	6mois	A
D	2 mois	b
E	4 mois	b
F	3 mois	d - a
Ω	0	c-e-f

Tableau 2.2 La table initiale d'ordonnancement T pour la réalisation d'un graphe potentiel-tâches [7]

Pour qu'une tâche puisse commencer, il est nécessaire que toutes les tâches qui la relient à la tâche du début **S** du projet, soient réalisées. On définit donc :

- **la date au plus tôt de la tâche**, qui correspond à la date de début, au plus tôt, de l'exécution de la tâche.

Exemple : 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 α (tâche de début du projet) et ω (tâche de fin du projet).

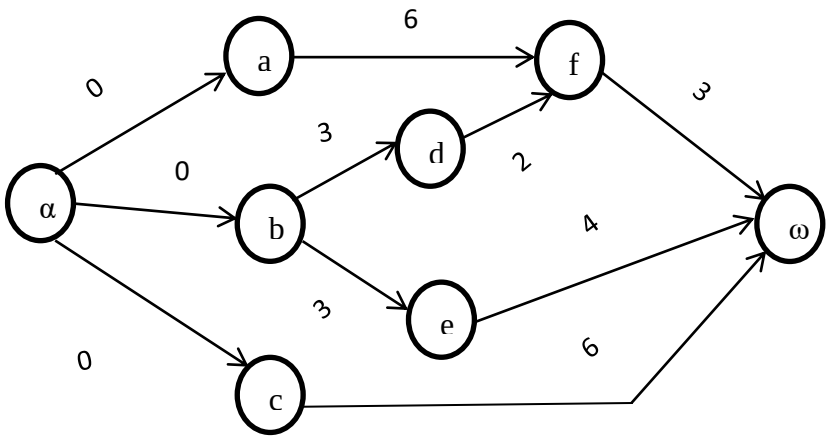


Figure 2.9 Graphe Potentiel-Tâches d'un ordonnancement [7]

2.5.3 Méthode PERT (Program Evaluation and Research Task)

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

2.6 Complexité des problèmes d'ordonnancement

D'une manière générale, les problèmes d'ordonnancement d'ateliers étant des problèmes d'optimisation combinatoires difficiles, il n'existe pas de méthodes universelles permettant de résoudre tous les cas. Plusieurs algorithmes peuvent être utilisés pour résoudre un problème d'ordonnancement mais tous ne sont pas équivalents. [11]

On distingue deux types de complexité: algorithmique et problématique [12]

2.6.1 Complexité algorithmique

Le temps et l'espace mémoire sont les paramètres déterminant la performance des algorithmes pouvant résoudre un problème donné. La performance d'un algorithme se mesure par rapport au temps de calcul nécessaire.

2.6.2 La complexité problématique

La complexité problématique dépend de la difficulté du problème à résoudre et du nombre des opérations élémentaires qu'un algorithme peut effectuer pour trouver l'optimum en fonction de la taille du problème.

Selon son degré de complexité, un problème peut appartenir à l'une des quatre classes suivantes [9] :

Les problèmes les plus difficiles : ce sont les problèmes indécidables pour lesquels il n'existe pas d'algorithme pour leur résolution.

Les problèmes de classe P : un problème de décision est dit facile où appartenir à la classe P s'il existe au moins un algorithme polynomial pour le résoudre. [12]

Les problèmes de la classe NP : Ce sont des problèmes NP-difficiles, qui ne peuvent à priori être résolus en un temps polynomial que par des méthodes approchées (heuristiques), **Les problèmes NP-Complets** : un problème de décision A est dit NP-Complet s'il appartient à la classe NP et si pour tout A' de NP :

- il existe une application polynomiale qui transforme toute instance I' de A' en une instance I de A.
- A' admet une réponse "oui" pour l'instance I', si et seulement si A admet une réponse "oui" pour l'instance I.

Autrement dit, s'il existe un algorithme polynomial pour résoudre A, alors, pour tout le reste des problèmes de la classe NP-Complets, il existe des algorithmes polynomiaux pour les résoudre. [9]

2.7 Comment résoudre les problèmes d'ordonnancement?

Au fil des années, de nombreuses méthodes de résolution de problèmes ont été proposées. Ainsi, une grande variété de concept et principe, de la stratégie et des performances ont été discernées. Cette variété et ces différences ont permis de regrouper les différentes méthodes de résolution de problèmes NP-difficiles en deux classes principales : la classe de méthodes exactes et la classe des méthodes approchées.

2.7.1 Méthodes exactes

Ces méthodes sont généralement utilisées pour résoudre des problèmes de petite taille. Dans ce cas, le nombre de combinaisons possibles est suffisamment faible pour pouvoir explorer l'espace de solutions en un temps raisonnable. On distingue trois sous-classes de méthodes exactes : la procédure de séparation et d'évaluation Branch and Bound, la programmation dynamique et la programmation linéaire.

2.7.2 Méthodes approchées

La résolution d'un problème d'optimisation combinatoire, de taille comparable à ceux rencontrés dans la pratique, se heurte à des tailles mémoire et des temps de calcul trop importants. L'objectif n'est plus alors d'obtenir systématiquement l'optimum mais plutôt d'obtenir une solution proche de l'optimum ou de « bonne qualité » en un temps minimal. Ainsi, au lieu d'effectuer une recherche exhaustive, les méthodes approchées échantillonnent l'espace de recherche, n'en considèrent qu'une partie, et fournissent ainsi, en un temps raisonnable, la meilleure configuration rencontrée.

On distingue deux types de méthodes : les heuristiques et métaheuristiques. [9][13]

2.7.2.1 Les heuristiques

Les heuristiques sont des méthodes empiriques basées sur des règles simplifiées pour optimiser un ou plusieurs critères. Le principe général de ces méthodes est d'intégrer des stratégies de décision pour construire une solution proche de l'optimum, tout en essayant de l'obtenir en un temps de calcul raisonnable. [9]

2.7.2.2 Les Métaheuristiques

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

L'ensemble des métaheuristiques proposées dans la littérature sont partagées en deux catégories : des métaheuristiques à base de solution unique et des métaheuristiques à base de population de solutions.

- **Les Métaheuristiques à base de solution unique**

Les métaheuristiques à base de solution unique débutent la recherche avec une seule solution initiale. Elles se basent sur la notion du voisinage pour améliorer la qualité de la solution courante. En fait, la solution initiale subit une série de modifications en fonction de son voisinage. Le but de ces modifications locales est d'explorer le voisinage de la solution actuelle afin d'améliorer progressivement sa qualité au cours des différentes itérations.

De nombreuses méthodes à base de solution unique ont été proposées dans la littérature, le recuit simulé, la recherche tabou. [13]

- **Les métaheuristiques à base de population de solutions**

Les métaheuristiques à base de population de solutions débutent la recherche avec une panoplie de solutions. Elles s'appliquent sur un ensemble de solutions afin d'en extraire la meilleure (l'optimum global) qui représentera la solution du problème traité. L'idée d'utiliser un ensemble de solutions au lieu d'une seule solution renforce la diversité de la recherche et augmente la possibilité d'émergence de solutions de bonne qualité.

Une grande variété de métaheuristiques basées sur une population de solutions a été proposée dans la littérature, algorithmes génétiques, et les algorithmes à base d'intelligence par essaims : l'algorithme d'optimisation par essaim de particules, l'algorithme de colonies de fourmis. [13]

La figure 2.10 [13] présente la classe de méthodes exactes et la classe des méthodes approchées.

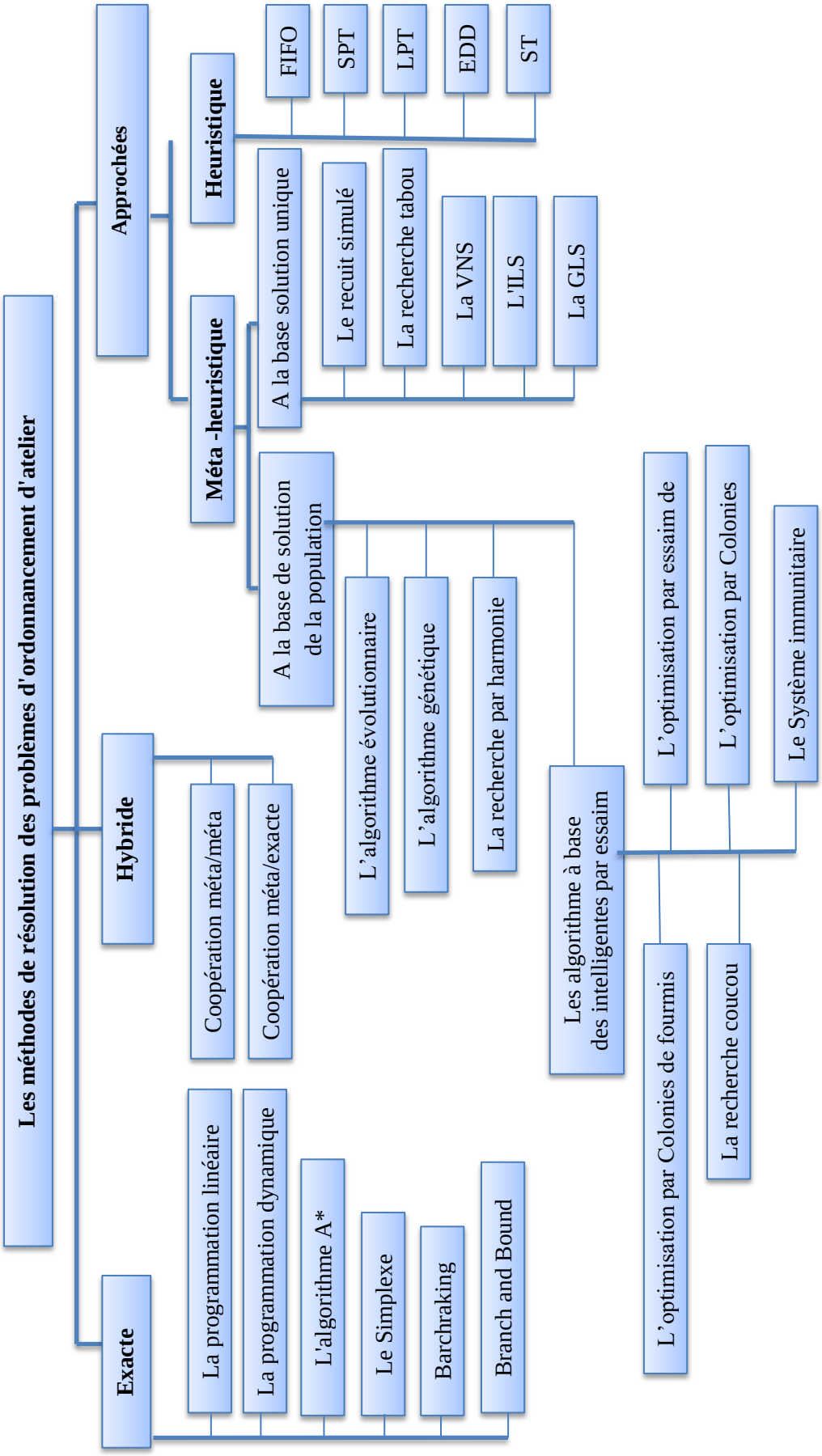


Figure 2.10 Classification des méthodes de résolution des problèmes d'ordonnancement d'atelier [13]

2.8 Problème d'ordonnancement 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 d'ordonnancement d'atelier. Le problème de job-shop généralise les problèmes d'ordonnancement tels que le problème à une machine, le flow-shop de permutation et le flow-shop général. Les problèmes de job shop se rencontrent, dans des entreprises manufacturières. Ces entreprises fabriquent, plusieurs produits hystérogènes. Chaque produit est caractérisé par une gamme de fabrication spécifique, connue à l'avance.

2.8.1 Description du problème d'ordonnancement job shop

Selon [2], Les problèmes d'ordonnancement d'ateliers de type job-shop sont les plus étudiés connus dans la littérature, et les plus difficiles à résoudre.

L'ordonnancement d'atelier consiste à optimiser l'exploitation des moyens de production limites (machines), pour réaliser un ensemble varie de produits (lots). La complexité ne réside pas dans le processus de fabrication prédétermine mais plutôt dans la combinatoire qui nait de la prise en compte des limitations de ressources.

Dans un problème d'atelier, un produit doit être fabrique sur différentes machines. Chaque machine est une ressource disjonctive, c'est-à-dire qu'elle ne peut exécuter qu'une tache à la fois, et les taches sont liées exclusivement par des contraintes d'enchainement. Plus précisément, les taches sont regroupées en n entités appelées travaux ou lots. Chaque lot est constitué de m taches à exécuter sur m machines distinctes.

L'ordonnancement job shop est associe a des lignes de production dédiées a la production de moyenne et de petites séries ou les changements de produits sont fréquents (généralement des produits de caractéristique différentes). Le job shop est considère comme étant un atelier a cheminement multiple. Chaque travail passe obligatoirement sur les machines selon une gamme fixée au préalable et spécifique à chaque travail. On parle, d'un atelier à flot multidirectionnel. Dans le cas où une opération nécessite l'exécution sur plus d'une machine on parlera d'un job shop a machines dupliquées, et si les opérations peuvent passer plus d'une fois sur une machine il s'agira dons ce cas d'un ordonnancement réentrant.

2.8.2 Définition formelle de problème d'ordonnancement job shop

2.8.2.1 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

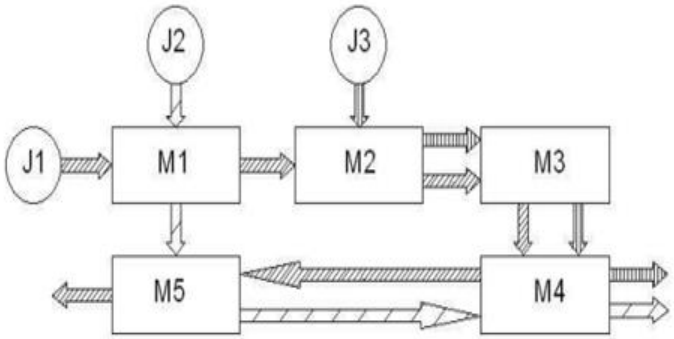


Figure 2.11 Problème de job shop classique composé de 3 jobs et 5 machines

2.8.2.2 Problème de job shop flexible

Selon [13], une façon d'augmenter la productivité d'un atelier est d'augmenter sa flexibilité, c'est de multiplier le nombre de machines qui réalisent la même tache. Le modèle résultant est connu dans la littérature comme un job-shop flexible. Dans ce modèle, les machines qui effectuent la même opération sont groupées dans un même étage.

Les problèmes de type job-shop Flexible (*FJSP*) sont une extension de deux problèmes d'ordonnancement : le problème de job-shop classique et le problème d'ordonnancement des machines parallèles. Le *FJSP* 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 2.2.

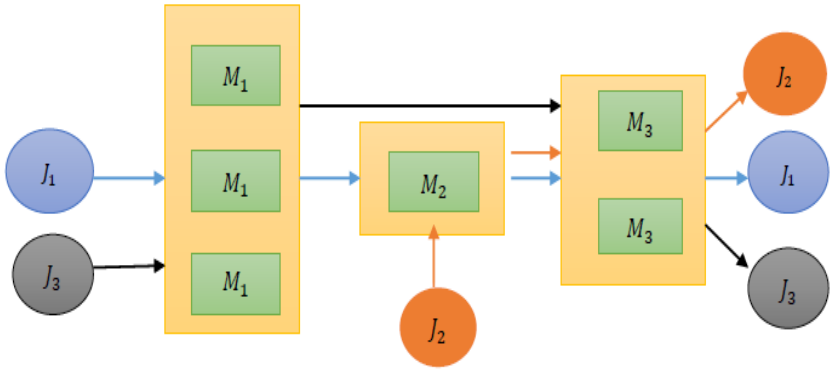


Figure 2.12 Représentation d'un système de type Job-shop flexible à deux étages [13]

Une remarque importante dans les problèmes de type job-shop flexible est que le nombre de machines peut varier d'un étage à l'autre, donc on distingue deux types de flexibilité de machines :

La flexibilité partielle (*P- FJSP*) : si chaque opération peut être exécuté sur quelque machines candidates, parmi l'ensemble de machines La flexibilité totale (*T- FJSP*) : si chaque opération peut être exécuté sur n'importe quelle machine (l'ensemble des machines candidates égal à l'ensemble globale des machines)

Les performances des machines ne sont pas forcément les mêmes pour toutes les machines d'un même étage. Ainsi on distingue :

Machines identiques : La durée d'exécution d'une tâche est la même sur toutes les machines.

Machines uniformes : La durée d'exécution d'une tâche est similaire sur toutes les machines.

Machines indépendantes : La durée d'exécution d'une opération dépend de la machine sur laquelle elle est exécutée.

2.8.3Complexité du problème d'ordonnancement job shop

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

De même, pour les problèmes a deux machines, dès qu'il y a recirculation, ils deviennent fortement NP-difficiles. L'expérience a montré que les problèmes de job shop ayant un

nombre m >2 machines, et optimisant les critères C_{max} et F sont NP-difficile au sens fort, même en utilisant des heuristiques.

Comme de donner une idée sur l'importance de l'espace de solutions pour une instance du problème, rappelons-nous qu'il existe (n!) m différentes solutions pour un problème de n tâches à réaliser sur m machines. Ainsi, pour un problème de taille de 10×10, il existe 39594×1065 solutions différentes (par comparaison, l'âge de la Terre ne dépasse pas 1018 secondes). Sachant que ce nombre ne comptabilise pas les ordonnancements semi-actifs.

Enumérer toutes ces possibilités pour arriver à la solution optimale n'est pas une chose réaliste. Il est à noter que ce nombre évolue plus vite en fonction de J qu'en fonction de M.

Autrement dit, l'ajout d'une tâche a beaucoup plus d'impact que l'ajout d'une machine. Par exemple :

▪ Pour un problème de 4 tâches et 5 machines (4×5) : $(n!)^m = 7962624$.

▪ Alors que, 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.

Pour résoudre un problème d'ordonnancement de manière efficace, il faut prendre en compte la particularité de chaque problème. C'est pourquoi il faut s'assurer de la classe de complexité associe au problème a ordonnancer. Il existe plusieurs méthodes de résolution du problème d'ordonnancement, la connaissance de la classe de complexité peut nous aider énormément quant au choix judicieux de la méthode de résolution.

2.8.4 Méthodes de résolution du problème de Job Shop

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

2.8.4.1 Les différentes méthodes de résolution

Une présentation de la totalité des méthodes et de leurs variantes apparaît une tâche difficile au vu du nombre d'approches proposées. Nous présentons les grandes orientations des

méthodes, en essayant de repérer les travaux de recherche essentiels.

Pour les Métaheuristiques, nous essayons de circonscrire avec plus de détails - sans prétendre toutefois être exhaustif - l'important des travaux effectués sur ces méthodes dans un paragraphe à part.

- Méthodes exactes :
  - Méthodes pour les problèmes polynomiau :- Le problème à deux machines.
    - Le problème à deux tâches.
  - La programmation mathématique.
  - Le branch and bound.
- Approximatives :
  - Les méthodes de relaxation.
  - Les méthodes de décomposition
  - Les heuristiques constructives : - L'approche par opérations.
    - L'approche par machines (Shifting Bottleneck).
    - L'approche par tâches.
  - Les métahruristiques : - Algorithmes Génétiques.
    - Recherche locale.
    - Le Recuit Simulé.
    - Recherche Tabou.

2.8.4.2 Les Métaheuristiques

Dès l'avènement des Métaheuristiques comme approche de résolution importante des problèmes difficiles offrant un bon compromis entre le coût de résolution et la qualité des solutions, le problème d'ordonnancement de Job Shop constitue l'un des domaines d'application de ces méthodes intensivement investigué. Bien qu'il existe une multitude de Métaheuristiques, trois ont en trouvé une place importante : Les Algorithmes Génétiques, Le Recuit Simulé et la Recherche Tabou.

Les Algorithmes Génétiques AG, sont largement appliqués aux problèmes d'ordonnancement de Job Shop. Leur application requiert les adapter au problème, en définissant adéquatement la représentation et les opérateurs.

2.9 Ordonnancement d'atelier en temps réel

Selon [5], Le problème d'ordonnancement en temps réel consiste à adapter en permanence les modalités d'exécutions d'ensemble des tâches par un ensemble des ressources à la situation réel du système considéré.

On rencontre souvent ce type de problème dans les ateliers de fabrication travaillant à la commande où le délai de livraison représente une des difficultés majeures. L'ordonnancement en temps réel offre à ces ateliers avec la prise en compte de ses caractéristiques réelles, une capacité de réagir aux déférents aléas (internes ou externes) auxquels ces ateliers sont soumis. Dans le domaine informatique la problématique de l'ordonnancement en temps réel revient à définir dans quel ordre exécuter les tâches sur chaque processeur d'une architecture donnée.
Cette problématique est en générale limité à une ressource ou plusieurs ressources en parallèle (processeurs), de plus l'interruption de tâches (préemption) est souvent autorisée contrairement aux ateliers.

2.9.1 Approches pour l'ordonnancement temps réel d'atelier

Pour la résolution des problèmes d'ordonnancement en temps réel deux types d'approches sont utilisées. Des approches basées sur un contexte statique et d'autres basées sur un contexte dynamique. Dans le contexte *statique* un ensemble des jobs est supposé connu à l'avance sur un horizon fini. La séquence de tâches est fixée par une approche prévisionnelle traditionnelle.
L'approche en temps réel consiste alors à contrôler le respect de la séquence proposée et lescdates de débuts prévues, tout en prenant en compte les perturbations apparues dans le système. Différents travaux ont été réalisés dans ce contexte. qui ont utilisés ce type d'approches en tenant en compte les nouveaux jobs inclus dans le plan de fabrication.
Dans le contexte *dynamique* les approches proposées considèrent que l'ensemble des jobs à ordonnancer ne sont pas connus d'avance, la solution ne s'effectue que dès qu'un job survient, l'ordonnancement donc consiste à affecter ses opérations aux ressources disponibles. Tuncel G. a proposé un ensemble des règles heuristiques de base pour l'ordonnancement dynamique des systèmes flexibles de production.

2.9.2 Méthode d'insertion de tâches

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

CHAPITRE 3

Les algorithmes génétiques pour la résolution de problèmes d'ordonnancement

3.1 Introduction

Dans cette section, nous donnerons le vocabulaire nécessaire à la compréhension des algorithmes génétique et nous identifierons les points importants de ces algorithmes, et l'application des algorithmes génétique pour la résolution du problème d'ordonnancement job shop.

3.2 Principe des Algorithmes Génétiques

3.2. 1 Historique

En 1859 les bases de l'évolution étaient posées par C. Darwin avec son idée de la sélection naturelle (« dans tout espèce les meilleurs sont sélectionnés »). En 1901 ce sont les bases de la génétique qui étaient posées par De -Vries suite à sa théorie du mutationnisme. En 1975 Jhon Holland proposa l'Algorithme Génétique. En 1989 Goldberg exposa les fondements mathématiques des algorithmes génétiques. [13]

Les algorithmes génétiques, techniques de recherche basées sur la théorie de l'évolution naturelle, et sur les mécanismes de la sélection naturelle et de la génétique. Ils se sont avérés très efficaces dans la résolution des problèmes complexes d'optimisation, tels que le problème du voyageur de commerce.

3.2.2 Présentation des algorithmes génétiques

Les algorithmes génétiques AG sont des algorithmes itératifs de recherche globale dont le but est d'optimiser une fonction prédéfinie appelée le critère ou fonction coût "fitness". Pour réaliser cet objectif, l'algorithme travaille en parallèle sur un ensemble de points représentant les solutions potentielles du problème. Cet ensemble est appelé population. Chaque point la constituant est appelé individu ou chromosome.

Un chromosome est une représentation ou un codage d'une solution d'un problème donné [99], il est constitué d'éléments, appelés gènes, dont les valeurs sont appelées allèles. [9]

Une première population est choisie soit aléatoirement, soit par des heuristiques ou par des méthodes spécifiques au problème, soit encore par mélange de solutions aléatoires et heuristiques. Cette population doit être suffisamment diversifiée pour que l'algorithme ne reste pas bloqué dans un optimum local. C'est ce qui se produit lorsque trop d'individus sont semblables. Les algorithmes génétiques génèrent de nouveaux individus, de telle sorte qu'ils soient plus performants que leurs prédécesseurs. Le processus d'amélioration des individus s'effectue par utilisation d'opérateurs génétiques, qui sont la sélection, le croisement et la mutation. [14]

Cinq éléments de base sont nécessaires pour l'utilisation des algorithmes génétiques [13]:

- une représentation génétique du problème, c'est-à-dire un codage de solutions utilisé sous la forme de chromosomes.
- Un mécanisme de génération de la population initiale. Ce mécanisme doit être capable de produire une population d'individus non homogène qui servira de base pour les générations futures. Le choix de la population initiale est important car il peut rendre plus ou moins rapide la convergence vers la solution optimale ;
- Une fonction à optimiser. Celle-ci prend ses valeurs dans R+ et est appelée fitness ou fonction d'évaluation de l'individu. elle est utilisée pour sélectionner et reproduire les meilleurs individus de la population. [13]
- Des opérateurs permettant de diversifier la population au cours des générations et d'explorer l'espace d'état. L'opérateur de croisement recompose les gènes d'individus existant dans la population, l'opérateur de mutation a pour but de garantir l'exploration de l'espace d'état.
- Des paramètres de dimensionnement : Taille de la population, nombre total de générations ou critère d'arrêt, probabilités d'application des opérateurs de croisement et de mutation. [15]

Le principe général du fonctionnement d'un algorithme génétique est représenté sur la figure 3.1.

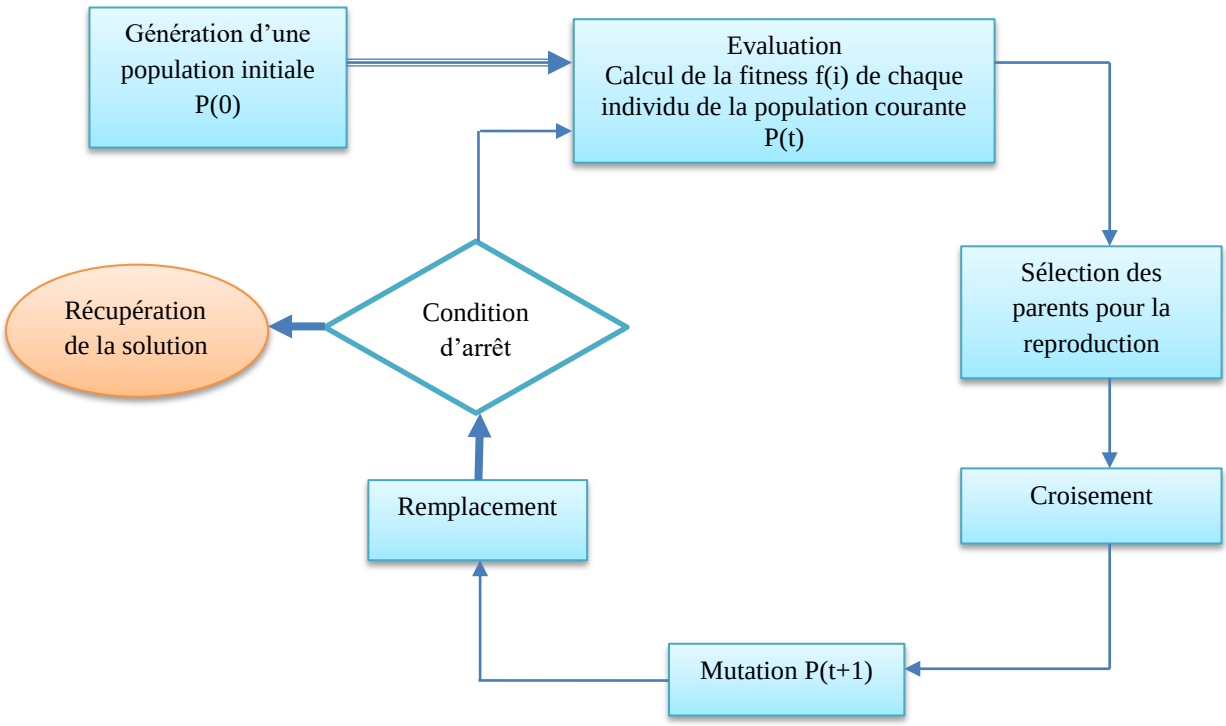


Figure 3.1 Fonctionnement général de l'algorithme génétique [13]

3.2. 3 Description détaillée

3.2.3.1 Codage des données

Chaque paramètre d'une solution est assimilé à un gène, toutes les valeurs qu'il peut prendre sont les allèles de ce gène, on doit trouver une manière de coder chaque allèle différent de façon unique (établir une bijection entre l'allèle "réel" et sa représentation codée). [13]

Le codage se base sur deux notions importantes :

• le génotype : représente l'ensemble des valeurs des gènes d'un chromosome.

• le phénotype : représente la solution du problème traduisant les données contenues dans le génotype.

Pour un passage immédiat entre le phénotype et le génotype, le codage est dit direct, sinon il est dit indirect et une procédure de passage est indispensable à définir. [9]

Dans la littérature, nous pouvons rencontrer trois grandes familles de codage d'une solution [13] :

- Le codage binaire : le codage binaire est le plus utilisé. Chaque gène dispose du même alphabet binaire {0,1}. Un gène est alors représenté par un entier long (32 bits), les chromosomes qui sont des suites de gènes sont représentés par des tableaux de gènes et les individus de notre espace de recherche sont représentés par des tableaux de chromosomes. Ce cas peut être généralisé à tout alphabet allélique.

1	1	0	1	0	0

Figure 3.2 Codage binaire d'un chromosome

- Le codage réel : ce type de codage est beaucoup plus efficace pour représenter des problèmes de type continu. Il représente les solutions par des suites de type réel, comme le montre la figure 3.3

56.25	1.3	5.62	32.8	-0.25	18.2

Figure 3.3 Codage réel d'un chromosome

- Le codage par valeurs : avec ce codage, chaque gène est codé par une valeur prise dans un ensemble fini ou infini. Ces valeurs sont bien entendu liées au problème à résoudre. [10][13]

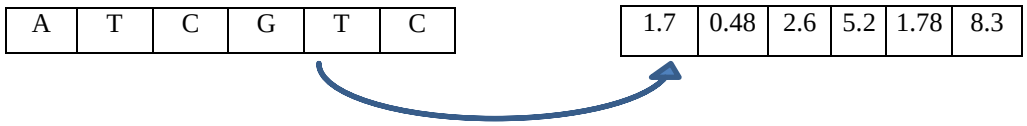


Figure 3.4 Codage par valeurs d'un chromosome

3.2.3.2 Génération aléatoire de la population initiale

Le choix de la population initiale d'individus conditionne fortement la rapidité de l'algorithme. Si la position de l'optimum dans l'espace d'état est totalement inconnue, il est naturel d'engendrer aléatoirement des individus en faisant des tirages uniformes dans chacun des domaines associés aux composantes de l'espace d'état, en veillant à ce que les individus produits respectent les contraintes . Si par contre, des informations a priori sur le problème sont disponibles, il paraît bien évidemment naturel d'engendrer les individus dans un sous-domaine particulier afin d'accélérer la convergence. Dans l'hypothèse où la gestion des contraintes ne peut se faire directement, les contraintes sont généralement incluses dans le critère à optimiser sous forme de pénalités. [15]

3.2.3.3 Evaluation : fitness

Une fonction d'évaluation est utilisée pour mesurer les performances de chaque individu, qui correspond à une solution donnée du problème à résoudre. Cette fonction permet d'évaluer la capacité d'un individu à survivre en lui affectant un poids appelé fitness. La force de chaque chromosome de la population est calculée afin que les plus forts soient retenus dans la phase de sélection, puis modifiés dans la phase de croisement et mutation. La fonction d'évaluation dépend d'un problème à un autre. Par exemple, dans le cas des problèmes d'ordonnancement, cette fonction peut prendre différents critères (makespan, retard, etc.)[14]

3.2.3.4 Sélection

La sélection permet d'identifier les individus susceptibles d'être croisés dans une population. Il existe plusieurs techniques de sélection. Nous présentons ici les quatre les plus utilisées parmi elles [14] :

- La sélection par la roulette (**RWS** : **R**oulette **W**heel **S**election) : Cette méthode exploite la métaphore d'une roulette de casino pour laquelle chaque individu de la population occupe une section de la roue proportionnelle à sa valeur d'évaluation. Ainsi, la position d'arrêt de la bille indique l'individu sélectionné.

La probabilité de sélection d'un individu x_i, notée P (x_i), définie par la relation [9] : N étant la taille de la population et f (x_i) l'évaluation ou fitness de l'individu x_i.

$$p(x_i)=\frac{f(xi)}{\sum_{k=1}^{N} f(xk)}$$

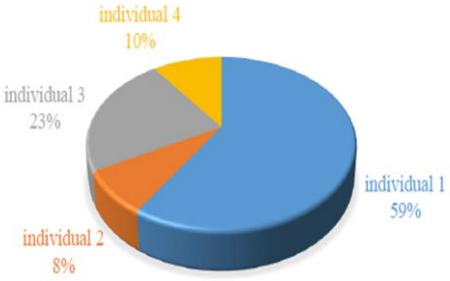


Figure 3.5 La roulette de sélection [13]

- Sélection par tournoi : le tournoi le plus simple consiste à choisir aléatoirement un nombre k d'individus dans la population et à sélectionner celui qui a la meilleure performance. Les individus qui participent à un tournoi sont remis ou sont retirés de la population, selon le choix de l'utilisateur. Avec le tournoi binaire, sur deux individus en compétition, le meilleur gagne avec une probabilité p ∈ [0, 5; 1]. [14]

3.2.3.5 Opérateur de croisement

Le croisement est un opérateur qui assure le brassage et la recombinaison des gènes parentaux pour former des descendants aux potentialités nouvelles, il correspond à un échange des matériels génétiques entre deux reproducteurs (parents) choisis d'une manière aléatoire parmi les géniteurs sélectionnés précédemment, pour former deux nouveaux chromosomes (ou enfants). L'opérateur de croisement étant aléatoire il se produit selon une probabilité P_c fixée par l'utilisateur en fonction du problème à optimiser.

Il existe plusieurs types d'opérateurs de croisement dont nous citons ci-dessous les plus utilisés [12]:

- Croisement avec un point : le croisement un point consiste à choisir un seul point de coupure, puis échanger les fragments situés après ce point de coupure. Comme le montre la figure 3.6 [13]

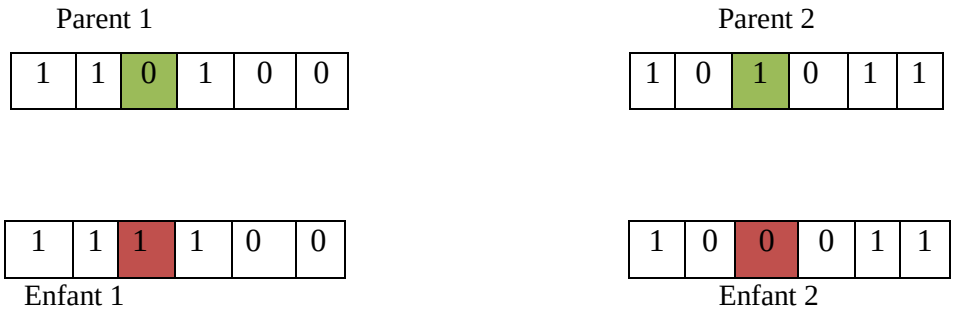


Figure 3.6 Croisement avec un point de deux chromosomes

- Croisement en deux points : ce type de croisement est utilisé en choisissant aléatoirement 2 points de coupure pour dissocier chaque parent en 3 fragments. Les 2 fragments en extrémités pour le Parent1 (respectivement Parent2) sont copiés à l'Enfant1 (respectivement Enfant2). On complète la partie restante de l'Enfant1 par les éléments du Parent2 et la partie restante de l'Enfant2 par les éléments du Parent1 en balayant de gauche à droite et en ne reprenant que les éléments non encore transmis. La figure 3.7 présente un exemple illustratif de ce type de croisement. [14]

Figure 3.7 Croisement avec deux points de deux chromosomes

- Croisement uniforme : Parmi les approches les plus connues du croisement uniforme, on cite l'utilisation du vecteur masque, le masque étant une suite aléatoire de 0 et 1 de la taille du chromosome. Il sert à déterminer de quel parent sont repris les gènes. La figure 3.8 présente un exemple du croisement uniforme. [9]

Parent 1	1	0	0	0	1	1	1	1	1	0
Parent 2	1	1	1	1	0	0	0	0	0	1
Masque	1	0	0	1	1	1	0	0	1	0
Enfant 1	1	1	1	0	1	1	0	0	1	1
Enfant 2	1	0	0	1	0	0	1	1	0	0

Figure 3.8 Croisement avec uniforme [12]

3.2.3.6 Opérateur de mutation

Cet opérateur consiste à changer la valeur allélique d'un gène avec une probabilité p_m très faible, généralement comprise entre 0.01 et 0.001.

Une mutation consiste simplement en l'inversion d'un bit (ou de plusieurs bits, mais vu la probabilité de mutation c'est extrêmement rare) se trouvant en un locus bien particulier et lui aussi déterminé de manière aléatoire; on peut donc résumer la mutation de la façon suivante :

Figure 3.9 Une mutation

On trouve plusieurs stratégies de mutation [13] :

- La mutation par échange : mutation par échange consiste à interchanger deux gènes sur un chromosome.
- La mutation par valeur Se fait d'une valeur donnée en une autre valeur déterminée, sur tous les gènes du chromosome.

3.3 Application des AG pour la résolution du problème d'ordonnancement job shop

Devant le succès grandissant des algorithmes génétiques dans le domaine de la recherche opérationnelle et de l'intelligence artificielle, leur efficacité à résoudre des problèmes d'optimisation très variés allant de l'économie à la biologie, en passant par l'informatique, la reconnaissance de formes et la commande des systèmes Ainsi, leur efficacité à s'approcher assez rapidement de l'optimum global et le pouvoir de donner contrairement à la plus part des autres méthodes existantes non pas une seule solution mais tout un ensemble, leur donne un avantage considérable et constitue une raison valable à leur utilisation malgré la concurrence féroce d'autre méthodes aussi efficaces qu'eux tels que le recuit simulé et la recherche tabou . Mais si la théorie des algorithmes génétiques est bien développée pour les problèmes qui peuvent être facilement codés par une chaîne binaire et sans ordre ou séquence de dépendance, elle ne l'est pas pour une grande partie des applications réelles et tout particulièrement pour les problèmes d'ordonnancement. [12]

3.3.1 Codage proposé

Le codage est la première étape à définir lors de l'utilisation des algorithmes génétiques. En effet, les problèmes tels que celui que nous traitons dans cette mémoire à savoir le problème d'ordonnancement de type job-shop

La littérature est assez riche dans ce domaine où on trouve une classification détaillée des différents types de codes existants. En considérant la classification don née par cette littérature abondante, on peut distinguer des représentations directes ou indirectes et qui peuvent être augmentées par des connaissances spécifiques au problème. [12]

3.3.1.1 Codage direct

Le chromosome représente une solution complète du problème traité. Ainsi, chaque chromosome contient toutes les informations utiles à la création de l'ordonnancement comme les dates de début des tâches et les contraintes de précédence. Dans ce cas, les opérateurs de croisement et de mutation sont adaptés pour ce type de codage afin de générer des solutions admissibles. Le codage basé sur les tâches et celui basé sur les opérations sont considérés comme des codages directs. [9]

Représentation par liste d'ordre/machine/date de début

Ce codage a été principalement développé pour une catégorie de problème d'ordonnancement plus restreinte où chaque ordre est limité à une seule opération qui doit être

exécutée par une seule machine. Chaque chromosome est une liste d'opérations uniques affectées chacune à une machine avec une date de début d'exécution.

Ordre A Ml Début A	Ordre B Ml Début B	…… ……. ……..

3.3.1.2 Codage indirect

Ce codage est plus souple que le précédent, il est utilisé dans la plupart des approches basées sur les algorithmes génétiques destinées à la résolution des problèmes d'ordonnancement. Dans ce cas, les chromosomes ne représentent plus directement un ordonnancement mais plutôt des listes de priorités sur une ressource qui permettent de déterminer un ordonnancement. La transition de la représentation du chromosome vers un ordonnancement légal doit être faite par une procédure de décodage, cette dernière doit garantir l'obtention d'un ordonnancement réalisable. Deux grandes catégories de codages peuvent être dégagées : ceux indépendants du domaine et ceux spécifiques au problème. [12]
Parmi les types de codages indirects, on cite :

Représentation spécifique au problème : Ce type de représentation enrichi le chromosome, les connaissances spécifiques au problème sont explicitement représentées dans le chromosome et de nouveaux opérateurs génétiques dépendants du domaine peuvent être conçus. Généralement on les obtient en ajoutant une ou plusieurs informations aux codages indépendants du domaine. Parmi ces représentations, on distingue :

Représentation par liste d'ordres/ gammes : Ce codage a été développé par Baghi. Dans ce cas, la gamme de chaque ordre est indiquée dans la représentation. La responsabilité du générateur d'ordonnancement est restreinte à la sélection des machines et des intervalles de temps.

Ordre 5 /Gamme B	Ordre 9 /Gamme C	Ordre 3 / Gamme A

Représentation par liste d'ordres /gammes /ressources : On ajoute à la représentation précédente les ressources sur lesquelles doivent s'exécuter les opérations on augmente les connaissances incorporée dans le chromosome

Ordre 7 Opér Bl Opér B2 Opér B3 M9 M3 M6	Ordre 9 Opér Al Opér A2 M6 Ml	…… …….. ……..

Cette représentation permet de diminuer considérablement la complexité des opérateurs génétiques à mettre en œuvre.

3.3.2 Génération de la population initiale

Après avoir choisi le codage, la deuxième étape consiste à déterminer la population initiale formée de solutions admissibles du problème. Il existe dans la littérature, plusieurs mécanismes de générations de la population initiale. Une méthode classique consiste à générer aléatoirement les chromosomes constituant la population initiale. Cette méthode permet d'avoir une population variée et d'explorer ainsi diverses zones dans l'espace de recherche. Aussi, pour générer la population initiale, des heuristiques peuvent être utilisées. Cette méthode peut, cependant, faire converger l'algorithme vers des optimas locaux.

Pour avoir un ensemble d'individus assez diversifié, la génération de la population initiale, dans notre application, se fait en générant des individus aléatoirement. La figure présente l'algorithme de génération de la population initiale. [9]

Algorithme : Génération aléatoire de la population initiale

Variable locale n : taille de la population

Début

  Pour i= 1 jusqu' à n

    Faire Générer un individu valide aléatoirement

  Fin pour

Fin

Algorithme de génération de la population initiale

3.3.3 Le croisement pour les problèmes d'ordonnancement

En passant au codage symbolique, la nécessité de concevoir de nouveaux opérateurs de croisement s'est fait considérablement sentir. En effet, l'imagination des chercheurs été à la hauteur de leurs ambitions et de nombreux opérateurs de croisement ont vu le jour, la littérature dans ce domaine est assez considérable.

Croisement d'ordre OX

Le croisement d'ordre OX (Order crossover) à été conçu par Davis. Deux zones aléatoires sont choisies. Les deux enfants sont obtenus en inversant les zones de croisement des parents. Les places restantes de l'enfant 1 et de l'enfant 2 sont remplies par les éléments non contenus dans leurs zones de croisement. On commence par le premier élément de la première zone du parent 1 (respectivement parent 2), on regarde si l'élément correspondant existe dans la partie construite de l'enfant 1 (respectivement l'enfant 2), si oui, passer à la position 2, sinon mettre

l'élément correspondant dans la première position en commençant par la zone 3 et on poursuit par la zone 1 de l'enfant 1 (respectivement l'enfant 2) jusqu'à ce que toutes les cases vides soient remplies. Ceci est illustré par l'exemple suivant :

Position	1	2	3	4	5	6	7	8	9	10
Parent 1	I	H	D	E	F	G	A	C	B	J
Parent 2	H	G	A	B	C	J	I	E	D	F
Enfant 1	F	G	A	B	C	J	I	H	D	E
Enfant 2	J	I	D	E	F	G	H	A	B	C
	Zone 1			Zone 2			Zone 3			

Croisement basé sur les positions UPX

L'UPX (Uniform Position crossover) Il a été également conçu par Syswreda. Un ensemble de positions est sélectionné d'une manière aléatoire, dans ce cas les valeurs correspondant aux positions choisies dans parent 2 (respectivement dans parent 1) sont copiées à la même position dans l'enfant 1 (respectivement dans l'enfant 2), les positions restées vides dans l'enfant 1 (respectivement dans l'enfant 2) sont remplies avec les valeurs manquantes en les prenant dans l'ordre du parent 1 (respectivement du parent 2). Ceci est illustré par l'exemple suivant [12] :

Position	1	2	3	4	5	6	7	8	9	10
Parent 1	I	H	D	E	F	G	A	C	B	J
Parent 2	H	G	A	B	C	J	I	E	D	F
Positions sélectionnées		*	*		*			*		
Enfant 1	I	G	A	H	C	D	F	E	B	J
Enfant 2	G	H	D	A	F	B	J	C	I	E

3.3.4 La mutation pour les problèmes d'ordonnancement

L'opérateur classique (modification aléatoire d'une valeur d'un gène) n'est pas applicable dans les problèmes d'ordonnancement. Les autres opérateurs de mutation (transposition ou inversion de deux gènes voisins ou aléatoire) peuvent être appliqués. Cependant, dans certains problèmes tels que le job shop, il faut tenir compte des contraintes de précédence et de ressources. Il est à noter aussi que plusieurs autres opérateurs de mutation ont été mis aux points. Parmi ceux-là nous pouvons citer les opérateurs de mutation avancée, ils utilisent des méthodes de recherche locale tels que le recuit simulé, la recherche tabou pour améliorer le résultat. [12]

CHAPITRE 4

Méthode d'insertion des nouvelles commandes

4.1 Introduction

Les problèmes de type job shop se rencontrent, dans des entreprises industriels. Ces entreprises fabriquent plusieurs produits et chaque produit est caractérise par une gamme de fabrication,

4.2 Système à étudier et hypothèses proposés

4.2.1 Organisation du système et modèle de transport

Dans ce travail, nous intéressons exclusivement à l'ordonnancement d'atelier de type job shop, cette dernière est classée dans les systèmes de production au flux discret. L'ordonnancement job shop on désigne sous le terme « job-shop J×M »: Ils sont composés d'un ensemble de *J* jobs (produit), chacun composé d'un ensemble de *nj* opérations qui peuvent être exécutées sur *m* machines. Les produits sont transportés à l'aide des chariots automatiques, qui peuvent contenir une ou plusieurs pièces (lot) du même produit (pour chaque chariot).

On examine que le système n'autorise pas les stocks intermédiaires. Alors le résultat immédiat est qu'une machine n'est pas disponible dès la terminaison d'une tâche. Celle-ci doit attendre que la machine suivante dans la gamme opératoire soit libérée pour lui transférer la pièce. De cette contrainte, on a emporté la propriété « succession sans interruption » qui impose toujours le non séparation entre tâches.

4.2.1.1 Les machines

Les machines représentent l'élément nécessaire du système, ce sont des équipements matériels réalisant des tâches spécifiques. Chaque machine est une ressource disjonctive, c'est-à-dire qu'elle ne peut exécuter qu'une seule tâche à la foi. On considère que la disponibilité de ces machines est toujours maintienne (pas de pannes toute au long de la phase d'exécution). Donc deux états caractérisent le fonctionnement des machines :

- l'état occupé ; la ressource est entrainé de réaliser une certaine tâche (état de marche).
- et l'état non occupé ou libre ; la ressource n'exécute aucune tâche (état de repos).

4.2.1.2 Les commandes

Le terme commande ici désigne le travail (job) à réaliser sur une matière première. Il contient l'ensemble des tâches nécessaires à la fabrication de certain produit. La réalisation de chaque commande est caractérisée par un délai de livraison qu'on veut respecter. Il est possible de dépasser ce délai dans le cas où la commande est non prioritaire.

Deux types de commandes sont alors tenus en compte :

- Des commandes déjà planifiées selon un ordonnancement prévisionnel, qui se répètent dans un cycle bien déterminé.
- Des commandes urgentes, se caractérisent par le fait que leur arrivé est aléatoire.

4.2.1.3 Les tâches

Une tâche est une entité élémentaire de travail localisée dans le temps par une date de début ou de fin, dont la réalisation est caractérisée par une durée défini. Les taches sont regroupées en n entités appelées travaux ou lots.

Un ensemble de propriétés, sur les tâches sont à souligner :

- La durée de chacune de ces tâches est déterminée par l'union des temps suivant :

- le temps de transport, c'est le temps de déplacement de la/les pièces vers la machine concernée.
- le temps d'exécution, qui représente le temps dans lequel une pièce doit rester dans la machine.
- le temps de préparation des ressources.

On note que s'il s'agit d'un lot, ce temps est calculé par la somme des temps de toutes les pièces de ce lot.

- La notion de préemption des tâche est non permit, c'est-à-dire que les tâches n'acceptent aucune interruption une fois sont commencées.
- La propriété succession sans interruption impose de ne pas avoir une séparation entre les tâches.

Dans certains cas, il est possible d'étendre la durée d'exécution d'une tâche, la ressource concernée n'est pas donc disponible pour exécuter une autre tâche pendant cette durée. Une opération ne peut être exécutée que sur une seule machine et sans interruption. [5]

4.2.2 Formulation du problème et notations utilisées

Les problèmes d'ateliers job-shop peuvent être formulés comme suit :

Un ensemble $J = \{1 \ldots n\}$, de n job (commande, produit). Certain job j est constitué de n_j tâches. O_{ij}^{uk} représente la $i^{ème}$ tâche du job j qui doit être exécutée sans interruption sur une unique ressource $k \in M$ dans l'ordre u.

Chaque tâche O_{ij}^{uk} est caractérisée par une date de début t_{ij}^{uk}, date de fin c_{ij}^{uk} et une durée d'exécution p_{ij}^{uk}.

Notations utilisée

MO_k : Ensembles des marges occupées sont des intervalles de temps où la machine k est en train d'exécuter une ou plusieurs tâches successives.

ML_k : Ensembles des marges libres sont des intervalles de temps où la machine k est en état de repos.

mo_{uk} : La marge occupée numéro u de la machine k

ml_{uk}: La marge libre numéro u da la machine k

dd_{uk} : Date début de la marge ml_{uk}

df_{uk} : Date fin de la marge ml_{uk}

j_a : Le job qui représente la nouvelle commande urgente

T_a : La date d'apparition de la nouvelle commande

Dl_a : Le délai de livraison de la nouvelle commande

n_a : Nombres des machine utilisées par la commande urgente

La Figure qui représente La notion des marges dans un exemple d'ordonnancement prévisionnel.

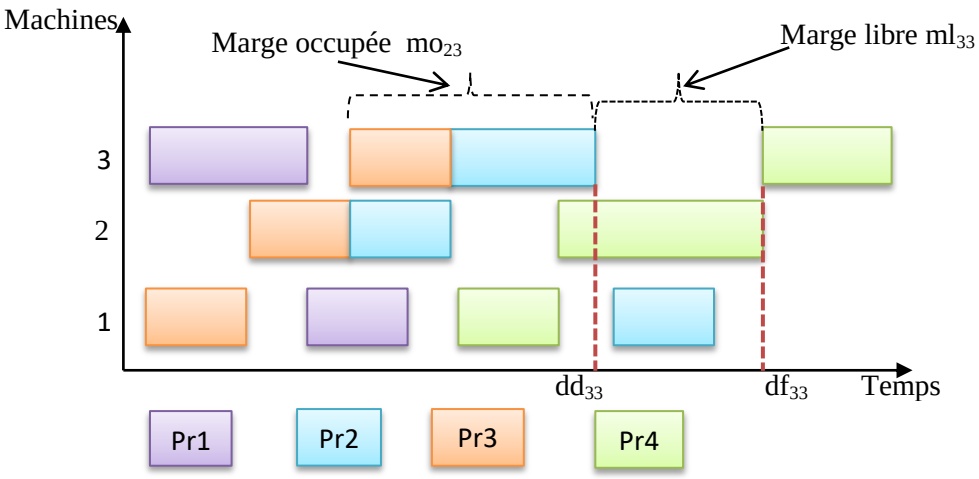


Figure 4.1 Diagramme de Gant d'un ordonnancement prévisionnel

À un instant donnée T_a de l'exécution de cet ordonnancement, une nouvelle commande urgente représentée par le job j_a de n_a tâches apparait.

Le problème imposé est comment insérer une nouvelle commande (la commande urgente) dans un système d'atelier de type job shop et la considérer comme prioritaire ?

4.3 Méthode d'insertion des nouvelles commandes

Pour la résolution du problème imposée, il y a une méthode qui consiste à générer un ordonnancement en temps réel et qui donne la possibilité d'introduire la commande urgente dans le plan prévisionnel.

L'insertion d'une commande urgente ne doit pas dépasser son délai de livraison il excite une solution avec possibilité de modifier le plan prévisionnel si c'est nécessaire.

La solution basée sur la recherche d'une position d'insertion admissible par la génération d'un ensemble des itérations. La spécificité de cette approche est que, pour chaque itération et dans le cas où la position définie n'est pas admissible, on doit effectuer une procédure de décalage qui sert à modifier l'ordre de passage existant, de tel sort que la position soit admissible. L'opération de la recherche est recommencée après une remise du plan à l'état initial. Un ensemble de solutions est donc généré à la fin de la recherche qui doit être arrêtée, soit par la détection d'une position admissible, soit par l'arriver au délai de livraison. Le but visé est de définir parmi ses solutions la bonne solution qui conduit à une modification minimale du plan initial. Deux critères sont utilisés pour sélectionner cette solution, la durée totale d'exécution C_{max} (Makespan).

4.3.1 Définition d'une position admissible

La détermination d'une position admissible, elle consiste à positionner les tâches de la commande urgente sur le plan prévisionnel par la détermination des dates début et de fin de chaque tâche, ensuite de faire subir cette position à un test d'admissibilité pour tester sa possibilité d'accepter les nouvelles tâches. Alors, l'obtention d'une position admissible implique l'arrêt de la recherche et maintenir cette solution. Dans le cas contraire celle-ci conduit au passage à la procédure de décalage. La sélection des tâches concernées par le décalage

4.3.2 Procédure de décalage

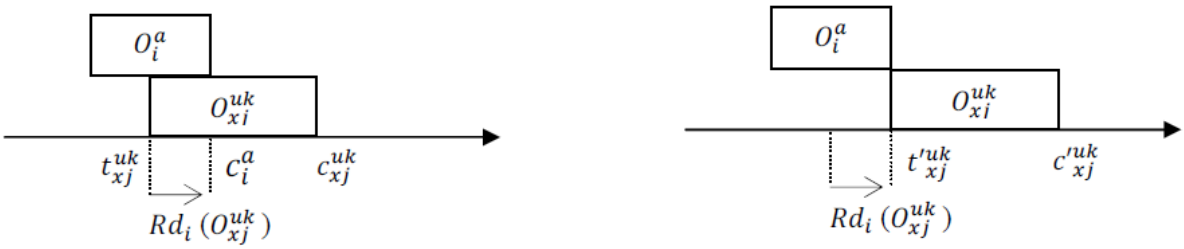
La modification du plan prévisionnel se réalise par l'opération de décalage d'une ou plusieurs tâches afin de générer des espaces libres supplémentaires pour qu'une position non

admissible devienne admissible. D'une autre manière c'est une quantité du temps additionnée aux dates début et de fin de ces tâches (Figure 4.2).

4.3.3 Création d'une position admissible et détermination du nouvel ordre

On définit par $\mathrm{Rd_i}(O_{xj}^{uk})$ le temps de décalage (retard) que doit subir la tâche O_{xj}^{uk} pour permettre l'insertion de la tâche O_i^a. Ce retard est déterminé par la différence entre la date début de la tâche concernée par le décalage et la date fin de la tâche à insérer (l'équation).

$$\mathrm{Rdi}\left(O_{xj}^{uk}\right) = c_i^a - t_{xj}^{uk}$$



a. Position avant le décalage

b. Position après le décalage

Figure 4.2 Exemple d'une opération de décalage effectué pour insérer la tâche O_i^a

La Figure 4.2 explique l'opération de décalage effectuée sur la tâche O_{xj}^{uk} où t'^{uk}_{xj} et c'^{uk}_{xj} représentent les nouvelles données de la tâche O_{xj}^{uk}, leurs calcule se fait par les relations suivantes:

$$t'^{uk}_{xj} = t_{xj}^{uk} + Rdi(O_{xj}^{uk}) \text{ et}$$

$$c'^{uk}_{xj} = c_{xj}^{uk} + Rdi(O_{xj}^{uk})$$

4.3.4 Choix du critère et définition de la bonne solution

La bonne solution est définie premièrement par la première position admissible trouvée avant que le délai de livraison n'est atteint et sans modification du plan prévisionnel, deuxièmement par la première solution qui donne une modification minimale du plan initial dans le cas où aucune position admissible n'est détectée. Cette modification touche généralement deux facteurs ; le facteur temps où la modification représente le changement des dates de livraison des commandes prévisionnelles (la satisfaction du client en termes de délai) et le facteur coût où la modification est traduite par le surcoût additionné au coût de production de ces commandes. Le critère sont alors retenus pour définir la bonne solution ; la durée totale d'exécution Makespan.

4.4.5 Durée totale d'exécution Maksepan

Pour ce critère la bonne solution est défini par la position qui donne un minimum du Makespan dont l'objectif est de minimiser le délai de livraison des commandes. Le calcul du Makespan, dans une itération, est effectué toujours à la fin de l'opération de décalage. La relation qui définit la valeur du Makespan est la suivante :

$$C_{max} = \max(max_{j=1}^{n}(c'_{njJ}), c'^{a}_{na})$$

c'_{njJ} : La date de fin de la dernière tâche du job j (déjà ordonnancé).

c'^{a}_{na}: La date de fin de la dernière tâche de la commande urgente.

4.5 Application de l'algorithme génétique dans l'insertion d'une tache dans un problème job shop

4.5.1 Le codage basé sur les opérations

Le codage basé sur les opérations représente un ordonnancement comme une suite d'opérations. Chaque opération est codée par un gène. La désignation de chaque opération peut se faire de deux façons : soit l'opération est désignée par un symbole qui la distingue de toutes les autres, comme le codage utilisé dans le problème du voyageur de commerce (TSP), mais qui est moins fiable pour le Job Shop Soit, toutes les opérations d'une tâche donnée portent le même symbole (le numéro de la tâche). L'interprétation de ce symbole se fait suivant l'ordre de son occurrence dans la liste. C'est cette dernière approche utilisée par Kebabla[10].

Avec ce codage, pour un problème n×m, le chromosome comprend n×m gènes. Chaque tâche apparaît dans le chromosome exactement m fois (nombre d'opérations).

Algorithme codage basé sur les opérations ;
Entrée : 1. $Ch[1,\dots,n\times m]$: un chromosome constitué de permutation de $\{J_1,\dots,J_n\}^m$;
2. J_1:$[O_{1,1}: p_{1,1},\dots,O_{1,m}: p_{1,m}],\dots, J_n$:$[O_{n,1}: p_{n,1},\dots,O_{n,m}: p_{n,m}]$: la gamme opératoire.
Sortie : • : un ordonnancement actif ;
Début
De $j \leftarrow 1$ **jusqu'à** $n\times m$
Faire
1. Déterminer la dernière opération ordonnancée de $Ch[j] = J_i$, soit $O_{i,k-1}$;
2. Insertion($O_{i,k}$) ;
Fin ;
Fin ;

Algorithme construction d'ordonnancement par codage basé sur les opérations [10]

La procédure "insertion" que nous avons utilisée à l'intérieur de cet algorithme, est détaillée

par l'algorithme

Algorithme Insertion($O_{i,j}$) ;
Entrée : 1. $O_{i,j}$: l'opération à insérer ;
2. M_k / $R(O_{i,j}) = M_k$: La machine demandée par $O_{i,j}$;
3. J_1:[$O_{1,1}$: $p_{1,1}$,...,$O_{1,m}$: $p_{1,m}$],..., J_n:[$O_{n,1}$: $p_{n,1}$,...,$O_{n,m}$: $p_{n,m}$] : la gamme opératoire.
Sortie : $M_k \oplus O_{i,j}$; où $\oplus$ désigne l'opérateur d'ordonnancement de l'opération $O_{i,j}$ sur M_k ;
Début
1. Déterminer $c_{i,j-1}$: fin d'exécution de $O_{i,j-1}$;
2. Trouver sur M_k une période vide p de début t / $p \geq p_{i,j}$, et $t \geq c_{i,j-1}$;
3. $t_{i,j} \leftarrow \max(t, c_{i,j-1})$; $c_{i,j} \leftarrow t_{i,j} + p_{i,j}$;
4. Mise à jour des intervalles vides et occupées de M_k ;
Fin ;

Algorithme Procédure d'insertion d'une opération dans un ordonnancement partiel [10]

4.5.2 Le croisement

Le croisement est un opérateur très important dans les algorithmes génétiques. Dans notre application, nous avons implémenté trois opérateurs de croisement

- Croisement avec deux points
- Croisement uniforme
- Croisement avec un point.

4.5.3 Opérateur de Mutation

Cet opérateur consiste à changer la valeur d'un individu par une autre valeur.

Mutation par échange

La réalisation de cette mutation se fait par le choix aléatoire de deux tâches, tous les gènes correspondant à l'une des tâches seront remplacés par l'autre.

Algorithme Mutation par permutation de tâches ;
Entrée : *Ch* : un chromosome de longueur *l* ;
Sortie : *Ch* muté ;
Début
1. *Choisir aléatoirement deux tâches à permuter* J_j *et* J_k ;
2. **De** $i \leftarrow 1$ **jusqu'à** *l* **Faire**
Début
2.1. **Si** $Ch[i] = J_j$ **Alors** $Ch[i] \leftarrow J_k$;
2.2. **Si** $Ch[i] = J_k$ **Alors** $Ch[i] \leftarrow J_j$;
Fin ;
Fin ;

Algorithme principe de mutation par permutation de tâches [10]

4.5.4 La sélection

La sélection est appliquée afin de favoriser au cours du temps les individus les mieux adaptés, à les faire se produire (duplication). Dans notre application, deux stratégies de sélection sont utilisées :

- La sélection par la roulette
- Sélection par tournoi

L'évaluation des individus

Le seul critère que nous avons utilisé pour l'évaluation des individus est le makespan de l'ordonnancement.

CHAPITRE 5

Conception et Réalisation

5.1 Introduction

Dans ce chapitre nous présenterons la réalisation de cette étude, nous avons conçu une application ayant pour but d'appliquer la méthode de résolution que nous avons présenté dans le précédent chapitre, à savoir l'algorithme génétique, adapté au problème d'ordonnancement temps réel d'atelier job shop.

5.2 Plateforme et outils de développement

5.2.1 Environnement matériel

Pour développer l'application, nous avons utilisé comme environnement matériel un PC hp qui possède comme caractéristiques :

- Un processeur Intel Core i3, 2.00 GHz.
- Une mémoire vive RAM de 4Go.

5.2.2 Environnement logiciel

Notre application a été réalisée en utilisant le langage de programmation C# sous le système d'exploitation Windows 10. On a utilisé l'environnement Visual studio 2013.

5.2.2.1 Microsoft Visual Studio

Microsoft Visual Studio est une suite de logiciels de développement pour Windows et mac OS conçue par Microsoft en 1997 La dernière version s'appelle Visual Studio 2017.
Visual Studio est un ensemble complet d'outils de développement permettant de générer des applications web ASP.NET, des services web XML, des applications bureautiques et des applications mobiles. Visual Basic, Visual C++, Visual C# utilisent tous le même environnement de développement intégré (IDE), qui leur permet de partager des outils et facilite la création de solutions faisant appel à plusieurs langages. La figure 5.1 présente l'interface de Visual Studio 2013.

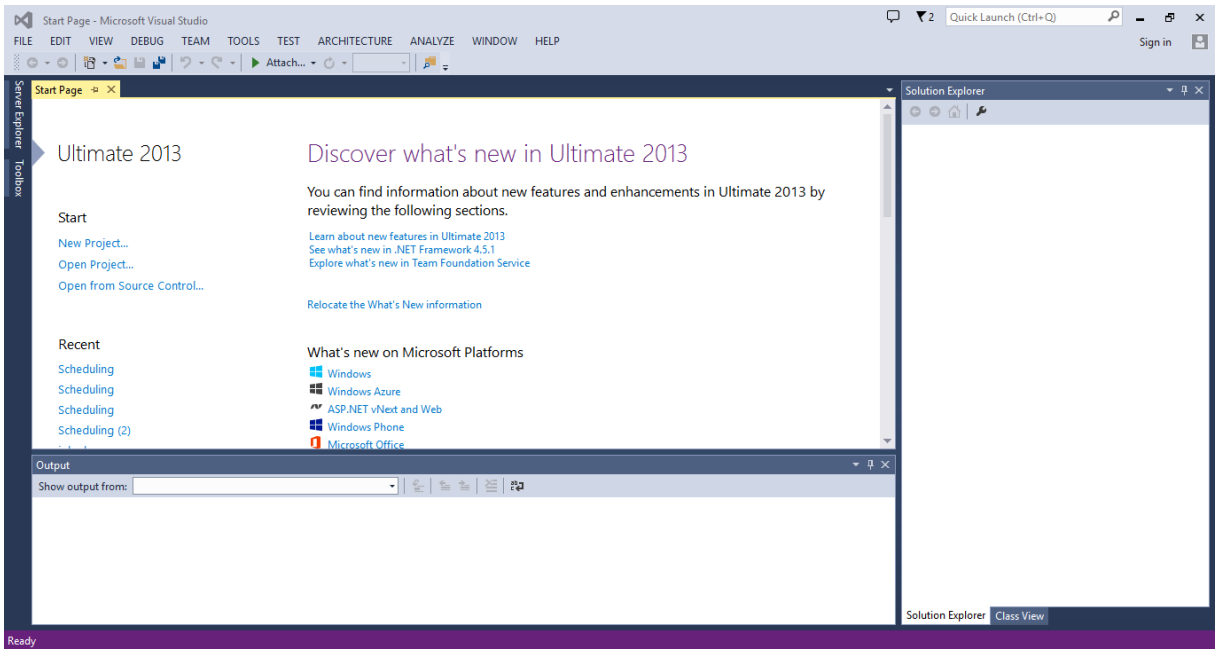


Figure 5.1 l'interface de Visual Studio 2013

5.2.2.2 C#

Le C# est un langage de programmation orientée objet crée en 2002 par Microsoft. Il a été créé afin que la plate-forme .Net possède un langage permettant d'exploiter toutes ces capacités. Au niveau de la syntaxe, le C# se rapproche beaucoup du Java. En tant que langage orienté objet, le C# prend en charge les concepts d'héritage, d'encapsulation et de polymorphisme. Les variables et méthodes sont encapsulées dans des définitions de classes. Une classe peut hériter d'une seule autre classe parente et peut implémenter plusieurs interfaces.

5.3 Interface graphique de l'application

Pour la résolution du problème d'ordonnancement temps réel d'atelier de type job shop, nous a conduits à développer un logiciel simplifié.

L'interface du logiciel conçue contient une seul fenêtre. Cette fenêtre permet à l'utilisateur de saisir les paramètres nombre des jobs et nombre des machines et nombre des opérations pour crée les tableaux, est choisi les paramètres de l'algorithme génétique.

Voici la page principale :

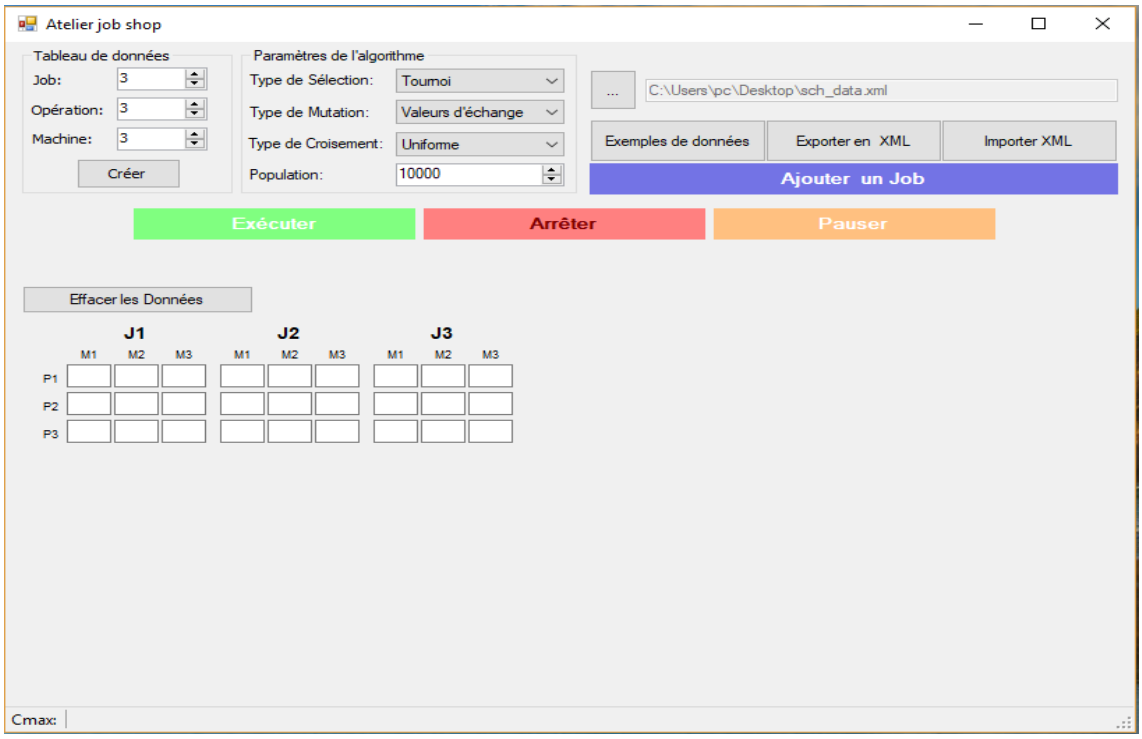


Figure 5.2 Interface graphique

La première étape d'utilisation est la détermination de l'instance de job shop constituant le problème à résoudre. Les instances doivent être de type Job Shop simple constitué de m machines et J jobs et p opérations.

La figure 5.2 illustré les instances

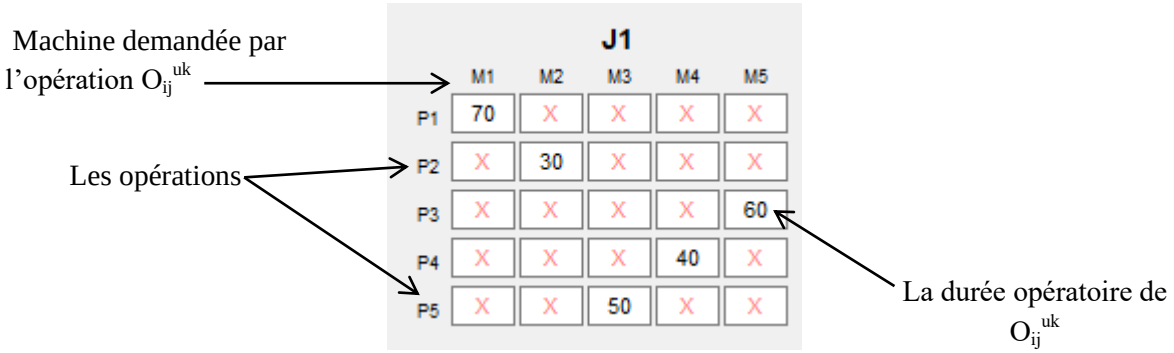


Figure 5.3 Format général des instances

Après détermination de l'instance-problème, on doit déterminer les paramètres de l'algorithme génétique :

- Choix de sélection :
  - Roulette.
  - Tournoi.
- Choix de croisement :
  - Croisement a un point.

- Croisement a deux points.
- Croisement uniforme.

- Choix de mutation :
  - Mutation par échange.
  - Mutation par valeur.
- Détermination de la taille de la population.

La figure 5.4 illustré un exemple de job shop composé de 5 jobs et 4 machines dont les gammes opératoires qui afficher dans les tableaux avec les durées opératoires

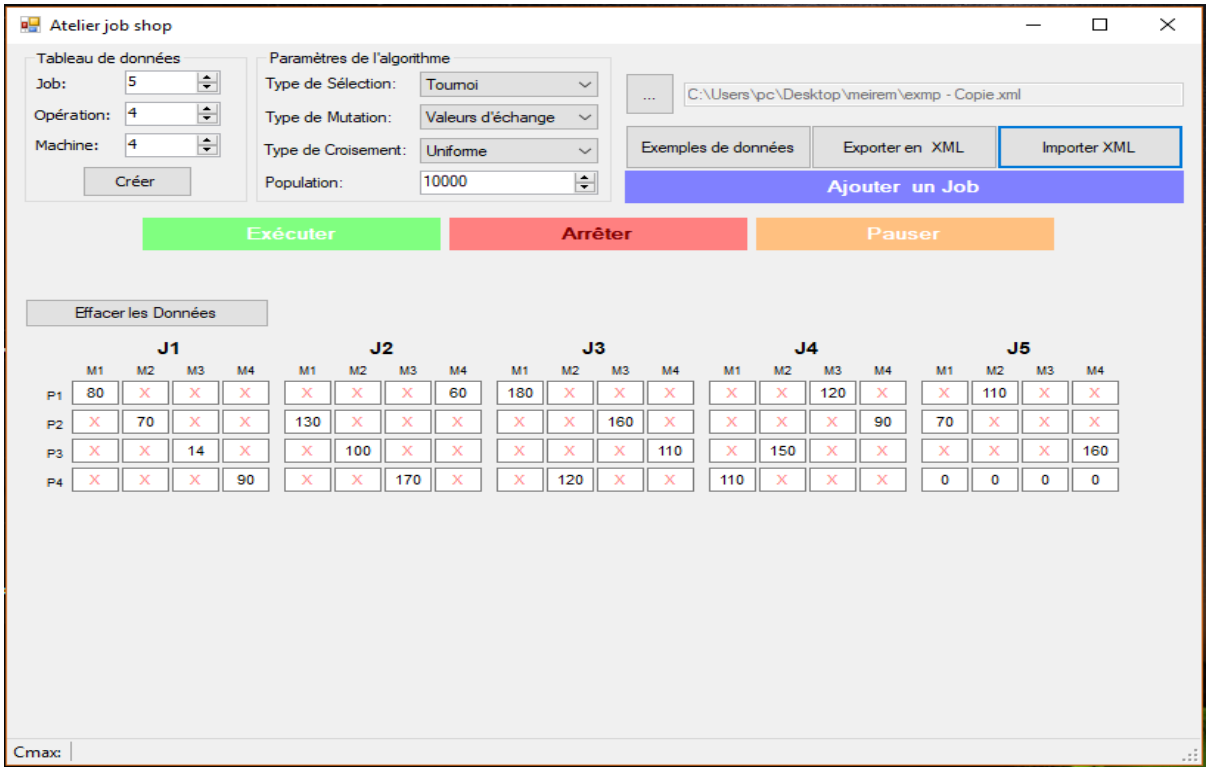


Figure 5.4 Interface graphique de l'exemple

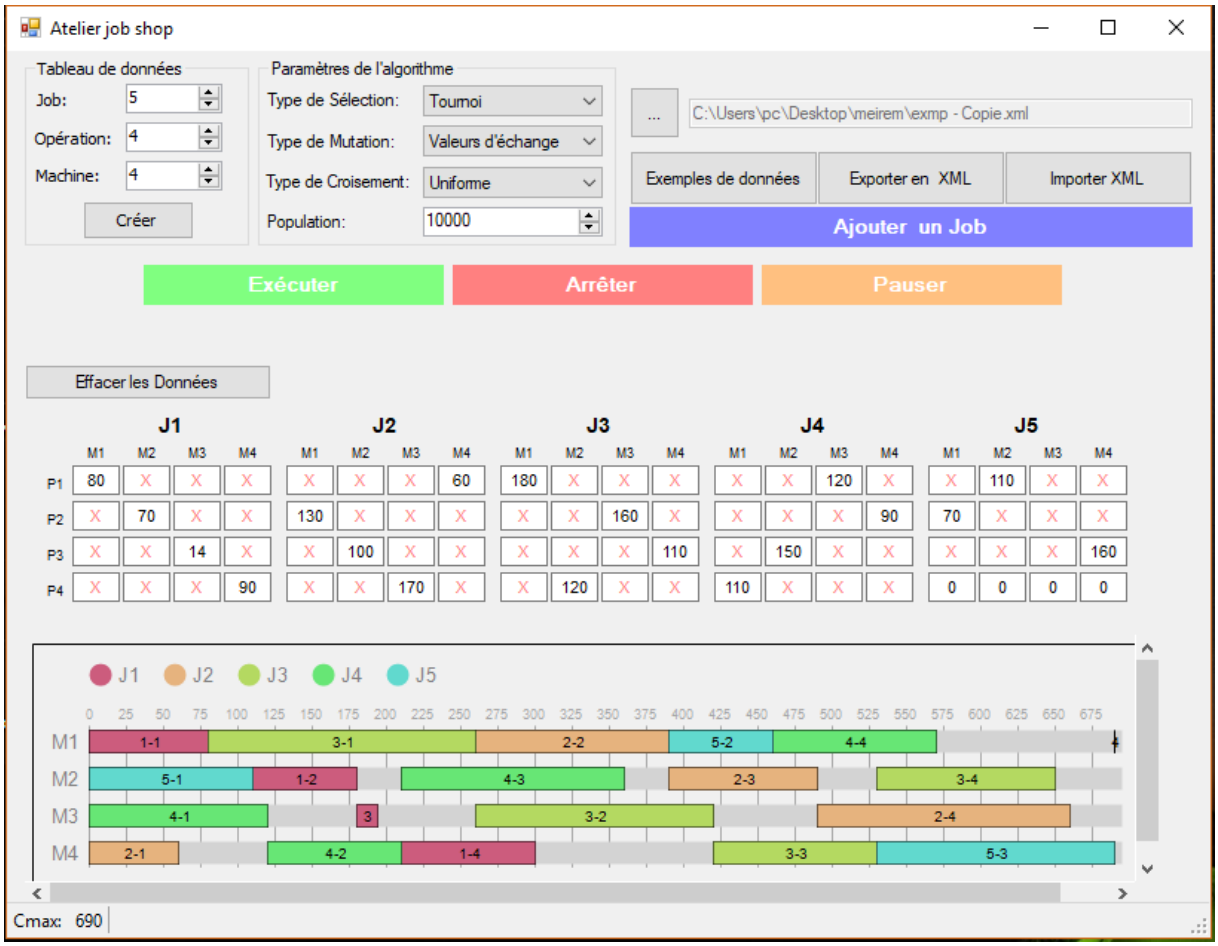


Figure 5.5 Interface après exécution le diagramme de Gantt représentant la solution optimale

Après l'exécution le résultat présenté le diagramme de Gantt (plan prévisionnel), et affiche aussi le Maksepan (C_{max}).

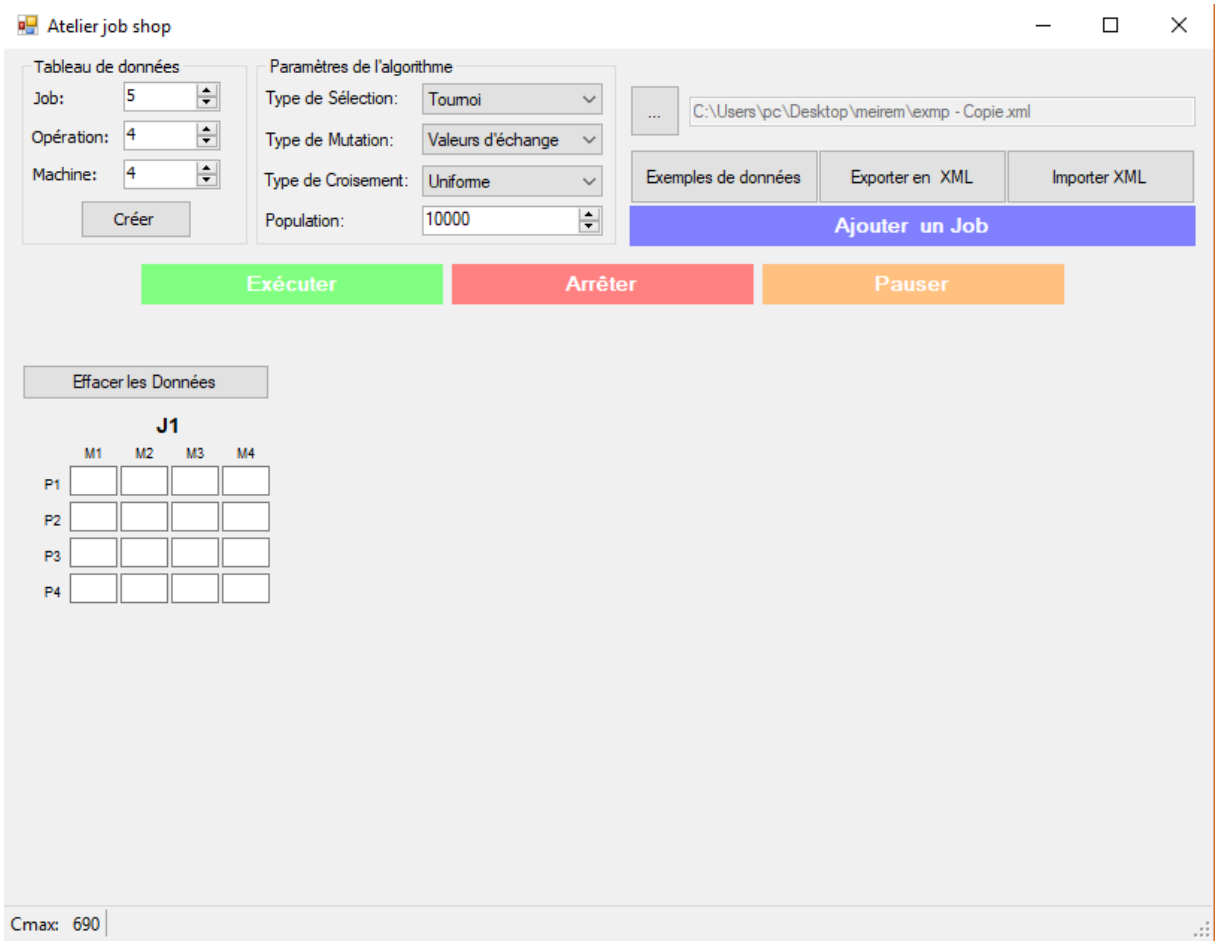


Figure 5.6 Interface pour ajouter nouvelle commande

Conclusion générale

Dans ce travail, nous avons abordé, les principales caractéristiques des problèmes d'ordonnancement de la production en générale et de manière plus particulière, les problèmes d'ordonnancement job shop. Ce problème est présenté par l'occurrence des nouvelles commandes qui doivent être insérées dans le programme de production courant de ce système.

Dans le premier chapitre nous avons abordé les systèmes de production, ainsi que la gestion de production.

Nous avons abordé dans le deuxième chapitre quelques concepts de base d'ordonnancement, des problèmes d'ordonnancement dans les ateliers de production,

Dans le troisième chapitre, nous nous sommes focalisés sur notre méthode de résolution, à savoir l'algorithme génétique et leur application sur le job shop.

Dans le quatrième chapitre, nous commençons par le problème d'ordonnancement job shop, suivi d'une description du problème à étudier.

Notre travail a permis permettant d'insérer les nouvelles commandes dans un ordonnancement existant.

Enfin le cinquième chapitre a été consacré à l'élaboration et l'implémentation de notre application.

- Comme perspective, nous envisageons de développer une autre approche de résolution basée sur un autre paradigme tel que la colonie de fourmis ou l'essaim de particules.
- tenir compte d'ordonnancement multi-objectif pour la sélection des solutions comme Makespan et coût de décalage.

Bibliographie

[1] G. Javel, Organisation et gestion de la production, 4ème édition, Dunod, 2010.

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

[3] A. Letouzey, Ordonnancement interactif basé sur des indicateurs : Applications à la gestion de commandes incertaines et à l'affectation des opérateurs, Toulouse, Doctorat, 2001.

[4] K. Tamani, Développement d'une méthodologie de pilotage intelligent par régulation de flux adaptée aux systèmes de production, Savoie, Doctorat, 2008.

[5] A. Hadri, L'ordonnancement par insertion en temps réel de la production dans un atelier flexible, Batna, Magister, 2012.

[6] M. Souier, Métaheuristiques pour la manipulation de routages alternatifs en temps réel dans un Job Shop, Tlemcen, Magister.

[7] H. Boukef ben othman, Sur l'ordonnancement d'ateliers job-shop flexibles et flow-shop en industries pharmaceutiques optimisation par algorithmes génétiques et essaims particulaires, Tunis, Doctorat, 2009.

[8] N. Mouhoub, Algorithmes de construction de graphes dans les problèmes d'ordonnancement de projet, Setif, Doctorat, 2011.

[9] A. Karray, Contribution à l'ordonnancement d'ateliers agroalimentaires utilisant des methodes d'optimisation hybrides, Tunis, Doctorat, 2011.

[10] M. kebabla, Utilisation des stratégies Métaheuristiques pour l'ordonnancement des ateliers de type Job Shop, Batna, Magister, 2008.

[11] A. Hassam ahmed, Développement et analyse de méthodes d'ordonnancement temps réel pour les systèmes flexibles de production, Tlemcen, Doctorat, 2012.

[12] K. Mesghouni, Application des algorithmes évolutionnistes dans les problèmes d'optimisation en ordonnancement de la production, Lille, Doctorat, 1999.

[13] I. Driss, Analyse d'un système job shop aspect ordonnancement, Batna, Doctorat, 2016.

[14] T. Chaari, Un algorithme génétique pour l'ordonnancement robuste : application au problème du flow shop hybride, Valenciennes, Doctorat, 2010.

[15] N. Durand, Algorithmes génétiques et autres outils d'optimisation appliqués à la gestion du trafic aérien, TOULOUSE, Doctorat, 2004.

ملخص

تعد مسائل الجدولة مجالًا واسعًا جدًا من فرع بحوث العمليات وإدارة الإنتاج. فهي موجودة في جميع قطاعات الاقتصاد و التصنيع و علوم الكمبيوتر.

يتناول هذا العمل مشكلة الجدولة الزمنية لورشات جوب شوب وتعمل هذه على عدة عمليات و عدة ألات .غالبية حالات الجدولة هي من التعقيدات الصعبة .و المشكل المطروح هنا يكمن حول كيفة ادراج طلبية جديدة مستعجلة في نظام مجدول و لحل هذا الاشكال استخدمنا الخورزميات الجينية.

الكلمات المفتاحية: الجدولة ، الجوب شوب ، الخوارزميات الجينية ، الوقت الحقيقي ، نظام الإنتاج.

Abstract

Scheduling issues are a very broad area of operational research and production management. Hey are present in all sectors of the economy, from manufacturing to computer science. This work deals with the scheduling problem of real-time Job Shop workshops with n tasks and m machines. Complexity studies have shown that the majority of job shop scheduling problem instances are NP-hard. The problem goal posed how inserted a new command into a forecast plan for the resolution of Problem uses the genetic algorithm.

Key words : Scheduling, Job Shop, Genetic Algorithms, Real Time, Production System.

Résumé

Les problèmes d'ordonnancement constituent un domaine très vaste de la recherche opérationnelle et de la gestion de la production. Ils sont présents dans tous les secteurs d'activité de l'économie, depuis l'industrie manufacturière jusqu'à l'informatique.

Ce travail traite le problème d'ordonnancement des ateliers de type Job Shop en temps réel, avec n tâches et m machines. Les études de la complexité ont montrés que la majorité des instances de problème d'ordonnancement job shop sont NP-difficiles. L'objectif de problème posé comment inséré un nouvelle commande dans plan prévisionnel pour la résolution de Problème en utilise l'algorithme génétique.

Mots clés : Ordonnancement, Job Shop, Algorithmes Génétiques, temps réel, système de production.

.