REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
الجمهورية الجزائرية الديمقراطية الشعبية

MINISTRY OF HIGHER EDUCATION
AND SCIENTIFIC RESEARCH

HIGHER SCHOOL IN APPLIED SCIENCES
--T L E M C E N--

وزارة التعليم العالي والبحث العلمي

المدرسة العليا في العلوم التطبيقية
-تلمسان-

المدرسة العليا في العلوم التطبيقية
École Supérieure en Sciences Appliquées

Mémoire de fin d'étude

Pour l'obtention du diplôme de Master

Filière : Génie industriel
Spécialité : Management industriel et logistique

Présenté par :
Abd-El-Krim GUERROUAZ
Mohammed El amine ABID

Thème

Optimisation de la tournée de véhicules avec contrainte de détérioration.

Soutenu publiquement, le 07/07/2022, devant le jury composé de :

M Mehdi SOUIER	Professeur	ESM. Tlemcen	Président
M Fouad MALIKI	MCB	ESSA. Tlemcen	Directeur de mémoire
M Mohammed DAHANE	HDR	Univ. Lorraine	Co- Directeur de mémoire
Mme Amina OHOUD	MCB	ESSA. Tlemcen	Examinateur
M Mohammed BENNEKROUF	MCA	ESSA. Tlemcen	Examinateur
Mme Latèfa GHOMRI	Professeur	Univ. Tlemcen	Invité

Année universitaire : 2021/2022

Remerciement

Au terme de ce travail nous tenons compte à exprimer nos sincère gratitude et profonde reconnaissance à :

Notre encadrant et chef de filière Mr Fouad MALIKI pour nous avoir formé durant ces trois années agréablement passées, ainsi que notre co-encadreur Mr Mohammed DAHANE de nous avoir permis cette opportunité et d'enrichir notre savoir sur le domaine, leur précieux conseils et orientations qu'ils nous avaient adressé durant toute la période de réalisation de ce modeste travail et nous vous sommes très reconnaissantes de bien vouloir porter intérêt à ce travail.

Aux membres de notre jury, pour le grand honneur qu'ils nous font en acceptant de juger ce modeste travail.

A l'ensemble de nos enseignants qui ont contribué dans la pertinence de notre formation.

Enfin, nous adressons nos remerciements à tous ceux qui ont participé de près ou de loin à la concrétisation de ce mémoire.

Dédicace :

Nous tenons à exprimer notre reconnaissance aux gens qui nous sont chères et qui ont contribués d'une quelconque façon dans ce modeste exploit, et que nous considérons comme facteur nécessaire dans notre réussite.

Nous dédions ce modeste travail à nos familles respectives pour avoir toujours été présents pour nous, pour nous avoir permis d'arriver à ce jour-ci où l'on espère les rendre fières après 19ans d'attention portée envers nous pour nous permettre les meilleures conditions d'études afin de s'épanouir dans notre domaine ; à Mounir, Annah, Foufa et Sid-Ali.

Résumé :

Dans ce projet, nous nous sommes intéressés au développement de trois méta-heuristiques à savoir les algorithmes génétiques, le recuit simulé et un algorithme hybride à partir des deux premiers, et ce afin d'optimiser la tournée de véhicules de livraison tout en considérant l'effet de détérioration qui retarde la date de livraison en ralentissant le véhicule chargé du transport. Nous résolvons d'abord un problème VRP classique, en suite nous développons un modèle de détérioration linéaire que nous inculquons au problème VRP, et pour finir, nous cherchons la position idéale pour une activité modificatrice de taux de rendement. Nous comparons notamment les solutions générées par nos trois algorithmes afin de déterminer quelle est la méta-heuristique la plus adaptée à ce problème.

Mots clefs : recherche opérationnelle, ordonnancement, logistique urbaine, transport, VRP, méta-heuristiques, algorithmes génétiques, recuit simulé, méta-heuristique hybride.

Abstract:

In this project, we are interested in developing three meta-heuristics, namely genetic algorithms, simulated annealing and a hybrid algorithm from the first two, in order to optimize the delivery vehicle routing while considering the deterioration effect that delays the delivery date by slowing down the vehicle in charge of the transport. We first solve a classical VRP, then we develop a linear deterioration model that we include in the VRP problem, and finally, we look for the ideal position for a rate modifying activity. Then we compare the solutions generated by our three algorithms in order to determine which meta-heuristic is best suited to this problem.

Keywords: operations research, scheduling, urban logistics, transportation, VRP, meta-heuristics, genetic algorithms, simulated annealing, hybrid meta-heuristics.

ملخص:

في هذا المشروع، ركزنا على تطوير ثلاث meta-heuristics: الخوارزميات الجينية، و simulated annealing وخوارزمية هجينة من الأولين، وهذا من أجل تحسين جولة مركبة التوصيل مع مراعاة تأثير التدهور الذي يؤخر تاريخ التسليم بسبب إبطاء السيارة المحملة بالنقل. نقوم أولاً بحل مشكلة VRP كلاسيكية، ثم نطور نموذج تدهور خطي نغرسه في مشكلة VRP، وأخيراً، نبحث عن الموضع المثالي لRMA . على وجه الخصوص، نقارن الحلول الناتجة عن خوارزمياتنا الثلاث من أجل تحديد علم التمثيل الغذائي الأنسب لهذه المشكلة.

الكلمات الرئيسية: البحوث التشغيلية، الجدولة، اللوجستيات الحضرية، النقل، VRP، meta-heuristics، الخوارزميات الجينية،hybrid meta-heuristics ،simulated annealing.

TABLE DES MATIÈRES.

LISTE DES FIGURES.

Chapitre 1 : généralités sur l'ordonnancement.

Chapitre 2 : généralités sur la Business Intelligence.

Chapitre 3 : application de la détérioration sur un VRP

LISTE DES TABLEAUX.

Introduction générale.

Introduction générale

De nos jours, produire un bien est à la portée de tous, ce qui a créé une offre plus considérable dans le marché, résultat les producteurs de bien et de services font face à un nouveau défi, en effet, en plus de la concurrence sur la qualité du produit, son prix, etc. avoir le meilleur produit sur tous ces critères ne servirait plus à grand-chose si notre système de livraison permettant de faire parvenir au client son besoin n'est pas performant ; d'autant plus que le coût de transport représente une grande part du prix d'un produit, il est donc judicieux d'optimiser son système de tel sorte qu'il soit rapide, à moindre coût tout en préservant le produit dans de bonne conditions pour préserver sa bonne qualité.

Etant donné l'importance du problème de tournées de véhicules (la logistique et transport représentent 95% du coût d'un bien), il est donc naturel que les entreprises portent plus d'intérêt intérêt à l'optimisation de son système, en l'occurrence, le développement de modèles permettant d'optimiser la tournée que suit sa flotte, voir même réduire le nombre de véhicules requis. Nous nous sommes intéressés à ce problème, tout en considérant la contrainte de la détérioration qui engendre l'accroissement du temps requis pour effectuer une livraison ; dans ce travail, nous travaillerons sur les données d'une entreprise de livraison de sachets de lait dans une zone urbaine, tel que nous partons du principe que la fatigue accumulée du chauffeur engendrera une baisse de la vitesse du véhicule, et par conséquent la livraison prendra tardera pour être effectuée.

La résolution de ce problème sera répartie en trois, en premier lieu nous allons résoudre un problème VRP classique, considérant trois véhicules de livraison, nous allons travailler sur un exemple de 21 points de livraison, le point de départ des véhicules est l'entrepôt de l'entreprise, l'aller et le retour des véhicule depuis et jusqu'à l'entrepôt sera comptabilisé notamment ; en outre, nous allons considérer l'effet de détérioration cité en haut, tel que nous assumons que la détérioration est une fonction linéaire qui croit à chaque fois que le véhicule livre un point de livraison ; pour finir, nous allons chercher la meilleure position de notre RMA qui est dans ce cas une pause permettant au chauffeur de se reposer, nous partons du principe que la détérioration sera réinitialisée après chaque RMA.

Nous proposons en premier lieu deux méta-heuristiques pour résoudre ces problèmes, nous optons pour les algorithmes génétiques pour sa pertinence prouvée lors de nombreuses études précédentes, et la méta-heuristique de recuit simulé parce que son programme est léger lors de sa compilation, d'autant plus que cet algorithme est facile à adapter pour tout type de problème. A la fin, nous allons développer un algorithme hybride, fusionnant les deux méta-heuristiques précédentes, et nous allons comparer entre elles pour savoir quelle est la plus adaptée à ce problème.

A cet effet, notre mémoire est subdivisé en trois chapitres, que nous allons présenter comme suite :

Introduction générale

Dans le premier chapitre, nous donnerons des concepts généraux sur l'ordonnancement, son domaine d'utilisation, les types de problèmes dont s'intéresse cette discipline, ainsi que les méthodes de résolution utilisées pour chaque problème.

Dans le deuxième chapitre, nous allons présenter un état de l'art sur les problèmes d'ordonnancement en général, puis sur les problèmes de détérioration assumant l'effet de détérioration, afin de clôturer ce chapitre, nous allons montrer l'apport de notre travail

Dans le troisième et dernier chapitre, nous allons parler du problème VRP classique, et de la logistique urbaine en général, en outre, nous procèderons à la présentation du problème qui fait office de sujet de notre projet, par la suite nous allons parler de notre méthode de résolution afin de mieux comprendre le travail fait, en suite nous allons commencer la résolution du problème, à commencer du problème VRP classique, pour enchainer après avec les modèle de détérioration cités en haut.

Chapitre I : généralités sur l'ordonnancement.

Chapitre I

I.1. Introduction :

Les entreprises d'aujourd'hui, faisant face à une concurrence de plus en plus rude, préconisent le développement dans le cadre de l'optimisation, et de cette politique régi l'amélioration des moyens et ressources dont dispose les entreprises à savoir les équipements (machines et outils sophistiqués), outils et méthodes de gestion (formations du personnel, gestion de la maintenance/qualité, gestion des stocks et logistique, etc.). En effet, dans les industries par exemple une optimisation de 0.1% reviendrait avec des avantages considérablement bénéfiques pour l'entreprise, à savoir, la gestion de la production qui permet d'augmenter le taux de production, réduire les coûts de la production, minimiser les temps…

L'ordonnancement de la production fait le sujet de nombreuses recherches pour l'amélioration dans le cadre de la gestion de la production, et les entreprises comptent sur l'ordonnancement au même instar que la planification, permet justement d'augmenter l'efficience de la planification, en assurant le meilleur ordre des tâches pour répondre aux commandes dans les délais à moindre coût.

I.2. La production :

La production au sens large, est un processus de transformation qui a pour but de réaliser un bien ou un service ; dans un sens plus restreint comme pour les industriels, "c'est une activité économique, qui vise à réaliser des biens ou des services, en exploitant des facteurs de production en l'occurrence les ressources de travail et le capital." [1]

I.3. La gestion de la production :

3.1 Définition :

Étant le pont qui lie entre l'amont (approvisionnement et stockage) et l'aval (stockage, commercialisation et distribution) des entreprises, "La gestion de la production peut être définit comme étant un ensemble d'activités participant à la **conception**, la **planification** des **ressources** (humaines, monétaires et des machines) nécessaires pour produire, leur **ordonnancement**, l'**enregistrement** et la **traçabilité** des activités de production, le **contrôle** des activités de production de l'entreprise." [2]

3.2 Le rôle de la gestion de la production :

D'après SAGE [3], géant de l'édition de logiciel diffères en l'occurrence la gestion intégrée, la gestion de production a pour but d'organiser et de fluidifier l'ensemble des étapes de la production industrielle. Les étapes du cycle de production sont au nombre de trois :

L'ordonnancement : génération et répartition des ordres de fabrication entre les machines, planification de la production.
La production : conduite des machines, management des opérateurs et gestion des stocks de matières premières.
Les contrôles et enregistrements : contrôle qualité, enregistrement des mouvements de stocks, enregistrement des temps.

3.3 Les objectifs principaux de la gestion de production :

"L'objectif essentiel, de la gestion de production, quelle que soit l'organisation est d'obtenir le produit permettant la satisfaction du client dans les délais à un coût et une qualité concurrentiels." [4]

I.4. Organisation de la production :

4.1 Typologie des systèmes de production :

Nous distinguons trois types principaux de la production : production en **continu** ; production en **discontinu** ; production par **projet**. Mais il est naturel de trouver d'autres types intermédiaires dans de nombreuses revues, par exemple Joan Woodward a proposé [21] en 1961 une typologie qui représentes quatre types :

1- Les productions de type **projet**, ce type ne permet pas la production d'une grande variété de produits, ni de produire une grande quantité, il permet par contre de réduire les stocks.

2- Les productions de type **atelier**, le cheminement des produits à travers les ateliers est très variable selon les personnalisations commandées par le client, ce qui permet une bonne variété de produits, pour des quantités modérées.

3- Les productions de type **masse**, inspiré par le taylorisme, ce type consiste en une chaine d'assemblage où les produits suivent tous un même cheminement et doivent passer par toutes les ressources, chaque ressources une seule et unique tâche tel que les mouvements superflus sont considérablement réduits à presque nuls, ce qui permet un gain de temps et par conséquent, de grande quantités de produits fini, pour une petite variété de produits.

4- Les productions de type process, basé principalement sur la redondance, le but est de maintenir le système en fonctionnement en cas d'occurrence d'une défaillance dans une installation, comme pour les raffineries où la production ne doit pas s'arrêter, ou dans les industries qui utilisent des fours tel que la re-fabrication d'un four peut aller jusqu'à des années (exemple : la révision d'un four à chaux prend une année). En plus de tout ça, la redondance active (mise en œuvre des moyens de production simultanément) permet aussi d'augmenter la capacité de production d'une usine, et donc réduire le temps par conséquent, mais cela va augmenter la quantité des stocks et des encours ce qui est généralement une contrainte en ce

qui concerne la gestion des entreprises, d'autant plus que la notion de variabilité (fluctuations à titre d'exemple) peut avoir des de grands effets néfastes lorsqu'on a une grande quantité d'encours à gérer [6], voir la loi de Little qui stipule que le nombre produits en cours (**WIP**) est égale au temps de cycle (**CT**) multiplié fois le taux de production (**TH**) $\mathbf{WIP} = \mathbf{CT} \times \mathbf{TH}$.

Taux de production (Throughput - TH) : pour une machine unique ou une chaine de production, le taux de production est la quantité de bon produits qui sortent par unité de temps.

Temps de cycle (Cycle Time - CT) : temps entre l'entrée du produit dans la chaine (début du routage) et l'arrive à la fin de la chaine (stock fin de chaine).

Encours (Work in Process - WIP) : quantité de produits à l'intérieur du système de production.

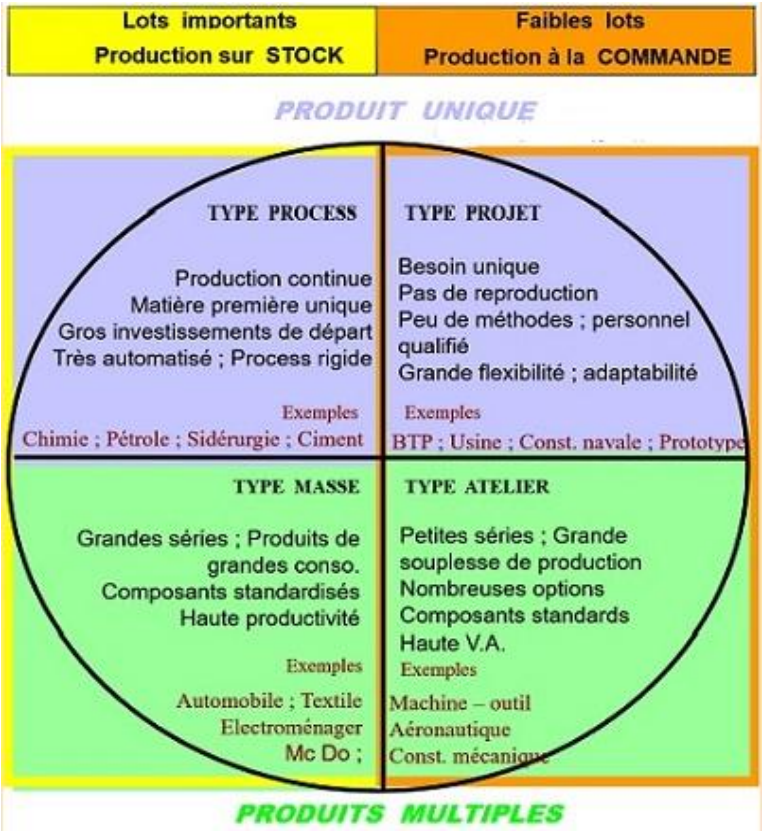


Figure 1 : organigramme sur les typologies de production. [5]

4.2 Les flux dans une entreprise :

Pour piloter son entreprise, et sa production par conséquent, il impératif de gérer ses flux, à savoir : les **flux financiers et administratifs** ; les **flux d'informations** (les échanges de données dans un service, les informations liés à la production, les

informations liés à la maintenance…) ; les **flux physiques** (quantité de produits en cours, quantité de matière première disponible et requise à la production, etc.), et ces trois flux forment le flux logistique qui représente lui aussi l'un des plus grand défi de la gestion des entreprises. Il existe deux types de flux logistiques :

4.2.1 Les flux externes : elle englobe les flux en amont et en aval de l'entreprise, soit, les flux d'approvisionnement (la circulation de matières premières depuis le fournisseur jusqu'à l'entrepôt), et les flux de distribution (concerne la circulation des produit finis/semi-finis depuis l'entrepôt de l'entreprise jusqu'au client final).

4.2.2 Les flux internes : quant à ce type, il représente les flux de production, il comprend le mouvement des différents composants et matériaux nécessaire pour la production, que ce soit dans la leur manutention, stockage ou transformation et fabrication.

Dans ce document nous allons nous intéresser seulement au deuxième type de flux (flux internes).

4.3 Les flux dans la gestion de la production :

Deux types de flux principaux ont vu le jour avec le développement de la gestion de la production :

4.3.1 Flux tiré « push »: dans son extrémité, ce type caractérise la politique de production du **Taylorisme**, les produits sont fabriqués en fonction de la demande prévisionnelle (la fabrication est lancée tenant compte de la quantité de produits en stock, et de la matière première disponible), pour garantir la disponibilité des produits à tout moment, cette méthode permet aux clients de ne pas attendre pour avoir son produit, et à l'entreprise de réduire le coût de production en produisant des quantités plus importantes, mais les coûts de stockages vont considérablement augmenter ce qui est généralement néfaste.

4.3.2 Flux poussé « pull » : opposément au flux tiré, ce type de flux stipule qu'il n'y aurait pas d'ordre de fabrication sans qu'il n'y ait de commande d'un poste client, ce permet de réaliser le zéro stock ; les entreprises utilisent cette méthode seulement dans le cas où le produit se fait sur commande (production de luxe) ou nécessite des personnalisations.

En outre, les industries sont passées vers un nouveau stade de gestion, qui est le Lean management, et donc de nouvelles façons de gérer les flux ont vu le jour :

4.3.3 Flux tendu : aussi appelé flux Just-in-time, il est la combinaison des deux flux précédents, il consiste à produire et fournir les produits juste à temps (minimiser les stocks sans faire attendre le client pour avoir son produit).

4.3.4 Flux synchrone : dans ce type d'organisation, on fournit aux ressources les différents besoins de la production (pièces, matière première, …) au fur et à mesure de la progression du processus de production, ils sont donc fournis juste au moment de leur utilisation.

Les deux flux tendu et synchrone, étant régis de la politique de production Lean, permettent de minimiser réduire les stocks et les encours ainsi que les coûts qui vont avec, établir plus d'organisation dans le lieu de production (pas de gaspillage de produits et d'espace), mais leur réalisation n'est pas aussi facile, il faudrait se doter de fournisseurs prêts à stocker et à fournir les besoins à tout temps d'une part, et d'une autre part, avoir en aval des commande régulières pour avoir un prévisionnel fiable.

En effet, il est judicieux d'avoir une gestion des flux de production de qualité en guise d'obtenir une performance globale de l'outil de production, plusieurs axes tels la planification de la production, l'optimisation des ressources et des procédés de production, etc. et à l'aide d'outils d'aide à la gestion de la production assistée par ordinateur en l'occurrence **ERP** (entreprise ressource planning) contribuent dans ce travail. Il est nécessaire de les maitriser dans l'ensemble, pour que l'on puisse dire que la production est bien gérée. Cependant il est impossible de traiter chacun de ces problème dans un seul travail de recherche, c'est donc pour ça que nous nous sommes investis dans l'étude de l'un des piliers de la gestion de la production, il s'agit de l'ordonnancement, une étude qui s'aligne toujours avec la planification de la production, d'autant plus que son importance est capitale au même instar que la planification, raison pour laquelle nombreuses entreprises parmi les plus grandes choisissent de réserver un service dédier à la planification et l'ordonnancement.

I.5. Généralités sur l'ordonnancement :

5.1 Définition :

Régi de la recherche opérationnelle, "ordonnancer, c'est programmer l'exécution d'une réalisation en attribuant des **ressources** aux **taches** et en fixant leurs dates d'exécution." [7] Autrement dit, l'ordonnancement est une organisation méthodique pour la détermination d'un plan optimal (ou réalisable) d'implantation.

5.2 Les tâches :

Ce sont des entités élémentaires de travail, définies par certaines caractéristiques (voir les références [7] et [20]), à savoir :

5.2.1 La date d'arrivée/disponibilité (release time, ready time ou arrival time) : notée $\mathbf{r_i}$, elle représente la date où la tâche i est prête à être traitée. Si toutes les taches ont la même date d'arrivée, on pose alors $\mathbf{r_i} = 0$.

5.2.2 Le temps/durée d'exécution (work, size ou execution time) : noté p_{ij}, il représente la durée de l'opération d'une tâche i dans une machine j.

5.2.3 La date de début : notée t_i, elle représente la date où commence le traitement de la tâche i, ceci dit, elle n'est pas toujours égale à r_i.

5.2.4 La date de fin (completion time) : notée c_j, elle représente la date de sortie (fin de traitement sur toutes les machines) d'une tâche j.

5.2.5 Date échue ou date au plus tard (due date) : notée d_j, indique que le traitement de la tâche i doit être accompli avant cette date.

5.2.6 Le facteur de priorité ou poids (weight) : noté w_i, il exprime la priorité relative de la tâche i.

5.2.7 Le temps d'installation (setup time) : il représente le temps requis pour préparer une machine au traitement d'une tâche.

Une tâche peut être récurrente (périodique) ou non récurrente ; aussi, elle peut être exécutée par morceaux (problème préemptif), tout comme elle peut ne pas l'être.

5.3 Les ressources :

Ce sont les moyens matériels dont on dispose, et qui sont nécessaires pour effectuer la production, disponibles à des quantités limitées, et des capacités supposées être constantes. Il existe cinq types de ressources, voir [7] :

5.3.1 Ressources renouvelables : ce type de ressources est disponible en même quantité après chaque utilisation (hommes, équipements, espace…)

5.3.2 Ressources consommables : ce sont les ressources dont la consommation globale au cours du temps est limitée (matière première, budget…)

5.3.3 Ressources doublement contraintes : leur utilisation instantanée et consommation globale limitées (source d'énergie, financement…)

5.3.4 Ressources disjonctives : ce type de ressources peut traiter une seule tâche à la fois.

5.3.5 Ressources cumulatives : contrairement aux ressources disjonctives, les ressources cumulatives peuvent traiter plusieurs tâches simultanément.

5.4 La différence entre l'ordonnancement et la planification :

Contrairement à la planification de la production, qui elle définit les quand et combien doit on produire chaque produit (quand doit on lancer un ordre de

fabrication par produit, et pour produire quel quantité), l'ordonnancement consiste à prioriser les différentes tâches et ordres **planifiés** à priori, ce qui revient à organiser dans le temps et de façon stratégique, une suite de tâches tenant compte des contraintes liées à la production. D'autres revues voir [12], font la distinction par les étapes que suit l'ordonnancement :

- La planification : définir les différentes opérations à réaliser, les dates correspondantes, les ressources à y affecter.
- L'exécution : mettre en œuvre les opérations définies l'étape précédente.
- Le contrôle : comparer entre les deux étapes précédentes (coûts et dates de réalisation).

5.5 L'objectif de l'ordonnancement :

Les buts d'un ordonnanceur sont variables selon le problème qui définit **la fonction objectif**, mais dans la généralité, il cherche à :

- Donner la part de temps nécessaire à la réalisation d'un processus en attente d'exécution,
- Utilisation optimale des ressources,
- Prendre en compte des priorités, etc.

Ces points sont parfois complémentaires (minimiser le makespan et minimiser le temps d'attente), parfois contradictoires (minimiser le coût, augmenter qualité et en respectant les délais) ; il est donc impossible de trouver à chaque fois une solution qui optimise tous ces objectifs, par conséquent les ordonnanceurs cherchent souvent le meilleur compromis à l'aide de méthodes diverses selon la problématique (méthodes de PERT, heuristiques d'ordonnancement, méta-heuristiques, …). Nous allons détailler quelques problèmes d'ordonnancement et les méthodes qui peuvent être utilisées dans ces cas-là plus tard dans ce chapitre. [9] [10] [11]

L'ordonnancement peut être appliqué dans une entreprise pour gérer sa production (dans ce cas précis nous appellerons les tâches par des **opérations**), comme il peut être appliqué pour gérer et piloter un projet (dans ce cas nous appellerons les tâches par **activités**). Vu que notre sujet porte sur la production, nous n'allons pas trop détailler sur l'ordonnancement pour la gestion et le pilotage de projets. D'autant plus que les plus grands défis en ce qui concerne l'ordonnancement se trouvent dans la production des entreprises, les méthodes d'ordonnancement pour la production sont souvent plus compliquées (préemption, pannes, recirculation, etc.) que celle dont fait face la gestion des projets (délais, antériorité, occupation), et ne sont pas nécessaires par conséquent dans la gestion des projets, tandis que les méthodes de la gestion des projets sont souvent utilisées pour trouver une solution (méthode **MPM** par exemple), ou pour visualiser sa solution dans le temps (diagramme de **Gantt**), sachant

que la visualisation dans le temps est très importante de nos jours pour piloter son entreprise dans une optique concurrentielle.

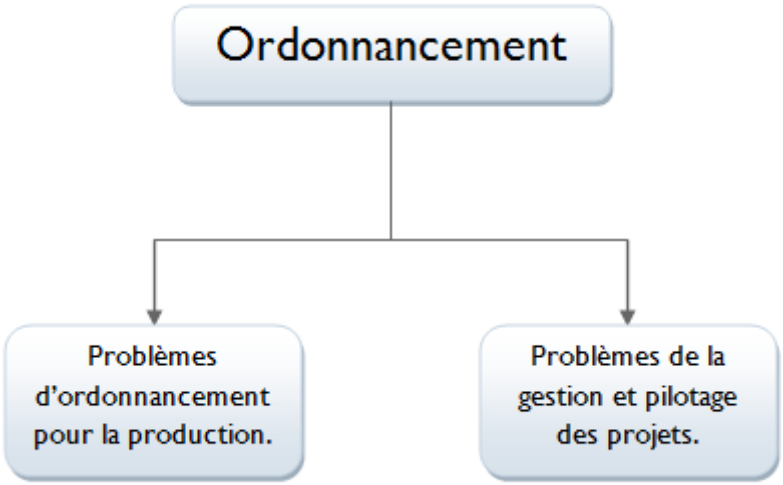
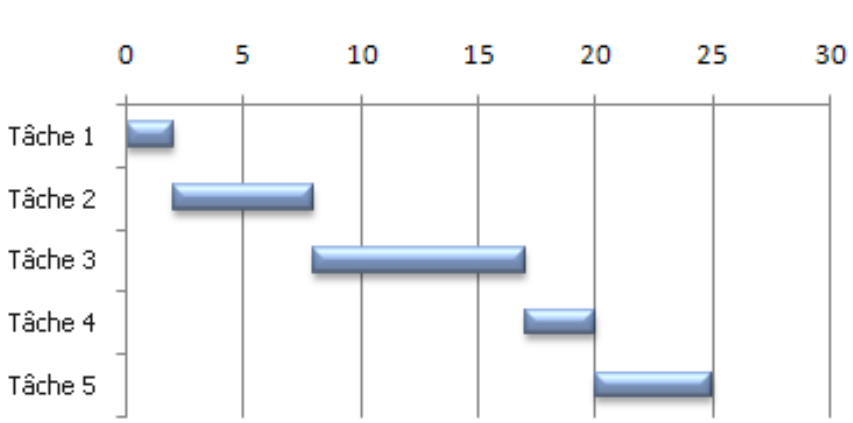


Figure 2 : les types de problèmes liés à l'ordonnancement.

5.5.1 Le diagramme de Gantt : mis au point pour la première fois par l'américain Henry Gantt, c'est un diagramme dans lequel on représente les ressources dans l'axe vertical et le temps dans l'axe horizontal, où chaque tâche sera représentée par un rectangle dont la longueur dépend du temps qu'elle occupe pour être réalisée ; tout comme on peut représenter les tâches sur l'axe vertical dans le cadre de la gestion des projets. [7] [12]

Figure 3 : exemple du diagramme de Gantt pour 5 tâches.



5.5.2 **La méthode des Potentiels Métra (MPM) :** développée par une équipe de chercheurs français, elle consiste à représenter les tâches par de sommets et les contraintes de succession par des arcs, dans chaque sommet on doit renseigner les dates où la tâche peut commencer (au plus tôt et au plus tard), et les dates où elle doit se terminer (au plus tôt et au plus tard) en plus du temps requis pour traiter la tâche, et dans chaque arc une valeur numérique qui représente un délai à attendre avant de passer au sommet qui suit.

Début au plus tôt	Tâche N°5	Fin au plus tôt
Début au plus tard	Durée de la tâche	Fin au plus tard

Figure 4 : représentation d'un sommet de la méthode MPM.

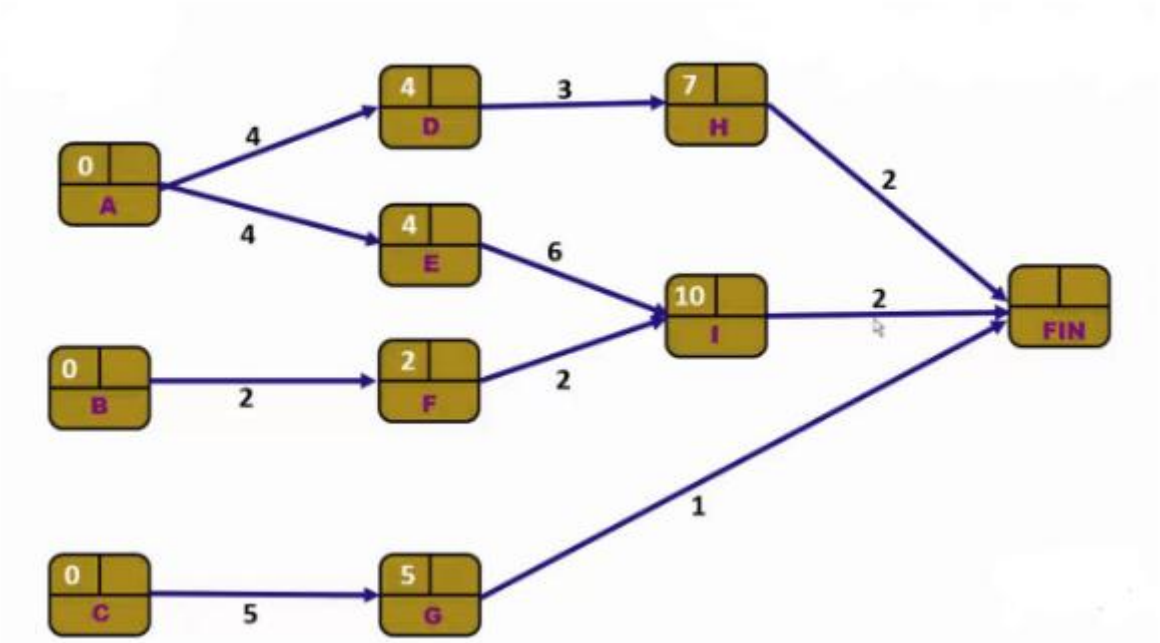


Figure 5 : exemple d'une MPM.

5.5.3 **La méthode P.E.R.T. (Program Evaluation and Research Task) :** cette méthode contrairement à la précédente, les tâches sont représentées sur les arcs, auquel nous associons un chiffre entre parenthèses pour représenter sa durée ; l'arc relie deux sommets, qui eux portent deux valeurs (la date au plus tôt et la date au plus tard), tel que le sommet qui se situe en amont de l'arc représente la date du début de la tâche et l'autre sommet représente la date de sa fin.

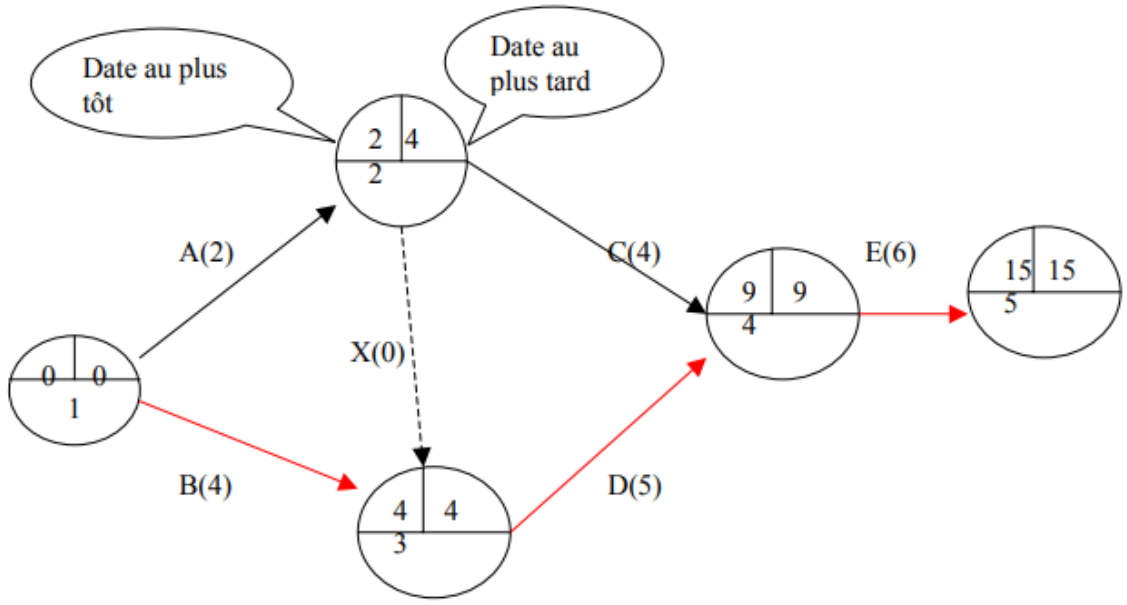


Figure 6 : exemple de la méthode P.E.R.T. [12]

I.6. L'ordonnancement dans la production :

6.1 Formulation des problèmes :

Il existe plusieurs façons de formuler un problème d'ordonnancement, nous utiliserons dans ce document la représentation en trois champs ($\alpha \mid \beta \mid \gamma$) introduite par Graham voir [8], tel que les trois champs représentent respectivement : l'environnement machines (contient obligatoirement une seule entrée), caractéristiques et contraintes dédiées à l'exécution des tâches (peut être vide dans le cas d'absence de contrainte, ou une concaténation de 1 à k entrées) et la fonction objectif (contient une seule entrée). [7] [19]

6.2 Le champ α (l'environnement machines) :

α prend sa valeur selon la typologie de l'atelier considéré, il existe cinq configurations (machines/ateliers) possibles : configuration basée sur le produit (taux de production élevé pour une faible variété de produits), configuration basée sur le process (diminue le risque d'arrêt du système en cas de défaillance des composants, et permet d'augmenter le taux de production des stations), configuration autour d'un

poste fixe (pour les projets de grande taille ou c'est au machines de se déplacer vers le produit, exemple : industrie navale, la construction de bâtiments et travaux publiques,…), configuration selon la technologie de groupes (permet une grande variété de produits selon le routage que prend ce dernier, mais par contre il est impossible de faire de grande quantités), configuration hybride[13]. Voici les typologies d'ateliers de production qui existent :

6.2.1 Cas d'une machine unique : le champ α prend alors la valeur 1 qui réfère au nombre de machines

Figure 7 : représentation d'un environnement machines pour α =1.

6.2.2 Cas des machines parallèles (Configuration basée sur le processus) : le champ α dans ce cas prend une des trois valeurs (**Pm**, **Qm**, **Rm**), tel que m représente le nombre de machines. Voir [7] [14]

Pm : m machines à vitesses identiques (machines identiques) et donc : $\forall i, p_{ij} = p_j$.

Qm : m machines à vitesses différentes (machines uniformes) et donc : $\forall i, p_{ij} = p_j/s_i$ où s_i est la vitesse de traitement de la machine M_i.

Rm : m machines différentes (machines indépendantes) et donc $\forall i, p_{ij} = p_j/s_{ii}$ Où s_{ij} est la vitesse de traitement de la tâche J_j par la machine M_i.

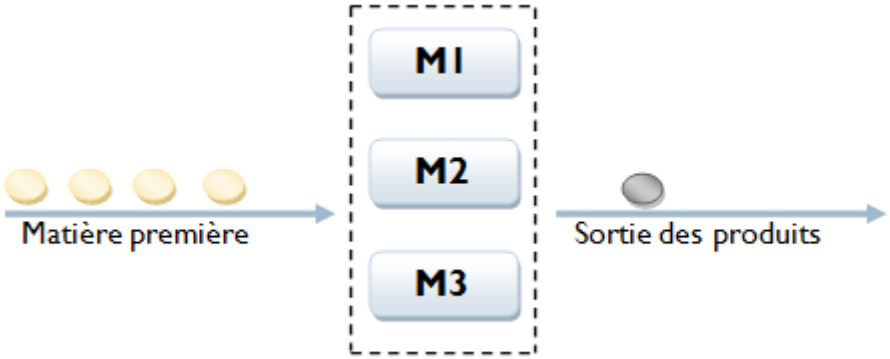


Figure 8 : exemple d'un système de production à machines parallèles.

Dans l'exemple illustré ci-dessus, nous avons trois machines en parallèle, et quatre unités de matière première qui attendent leur traitement, la première unité de MP va rentrer dans la première machine, la deuxième unité de MP va rentrer dans la deuxième machine, et troisième dans la troisième machine, ainsi, la quatrième unité de MP doit attendre que l'une des machines se libère pour qu'elle puisse rentrer se faire traiter dedans.

6.2.3 Cas des machines en séries (Configuration basée sur le produit) : dans ce cas l'atelier est à flux continu, et le champ α prend une des deux valeurs (Fm ou FFs) pour le flow shop et le flow shop flexible respectivement. Dans un atelier flow shop, le défi n'est pas de trouver le meilleur routage des produits, mais plutôt d'ordonnancer les tâches sur chaque machines (qui passe en 1er et qui passe en dernier), les tâches doivent passer par conséquent par toutes les machines, dans le cas où l'ordre d'exécution est le même pour toute les machines, donc l'accès aux machines se fait en **FIFO** : nous pouvons dire alors que c'est un flow shop de permutation ; quant flow shop flexible, il est un cas particulier du flow shop de permutation, tel qu'il représente un nombre de s stations en cascade (en série), où chaque station est composée de machines en parallèle (les tâches peuvent être traitées par n'importe quelle machines dans la station), ce type d'ateliers est aussi appelé atelier à cheminement unique avec machines en exemplaires multiples. [7] [15]

Figure 9 : atelier Flow Shop à m machines.

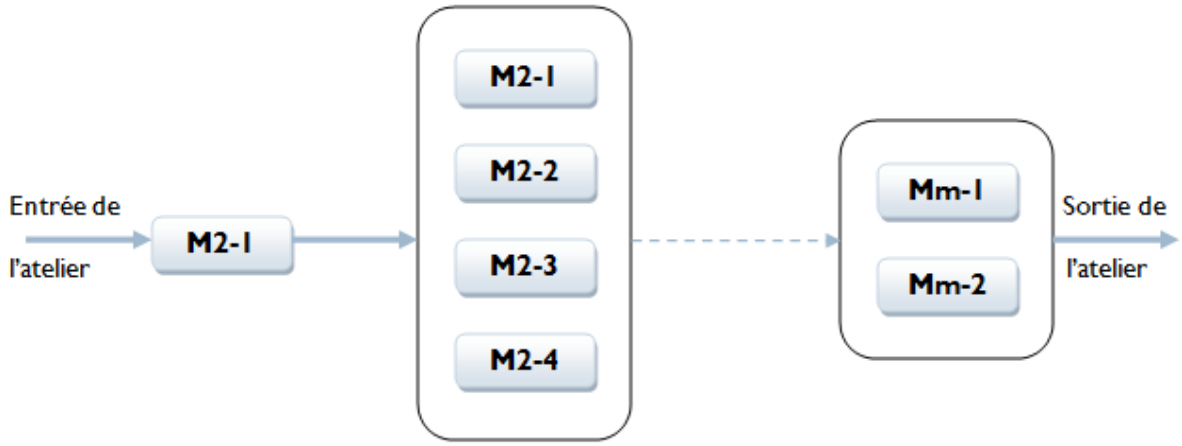


Figure 10 : atelier Flow Shop flexible à m stations.

6.2.4 Cas d'un open shop (Configuration Hybride) : dans les ateliers à cheminement libre chaque tâche doit passer par toutes les machines, mais le temps opératoire d'une tâche peut être nul dans certaines machines, c'est à l'ordonnanceur de fixer le routage de chaque tâche, sachant qu'il n'y a pas de restrictions sur le routage, [7] dans ce cas le champ α prend la valeur (**Om**).

6.2.5 Cas d'un job shop (Groupe technologie layout) : la technologie de groupe consiste à regrouper dans un seul endroit les composants (machines) en familles selon leurs caractéristiques communes, et ce, en guise de gain de temps en réduisant les déplacements lors de la fabrication des pièces qui nécessitent le même traitement. Dans ces ateliers à cheminements multiples, chaque tâche possède un cheminement unique, et elle peut passer par une machine/station zéro ou une seule fois, tout comme elle peut repasser plusieurs fois (recirculation). Dans ce cas le champ α prend la valeur (**Jm**). [7] [15] [16]

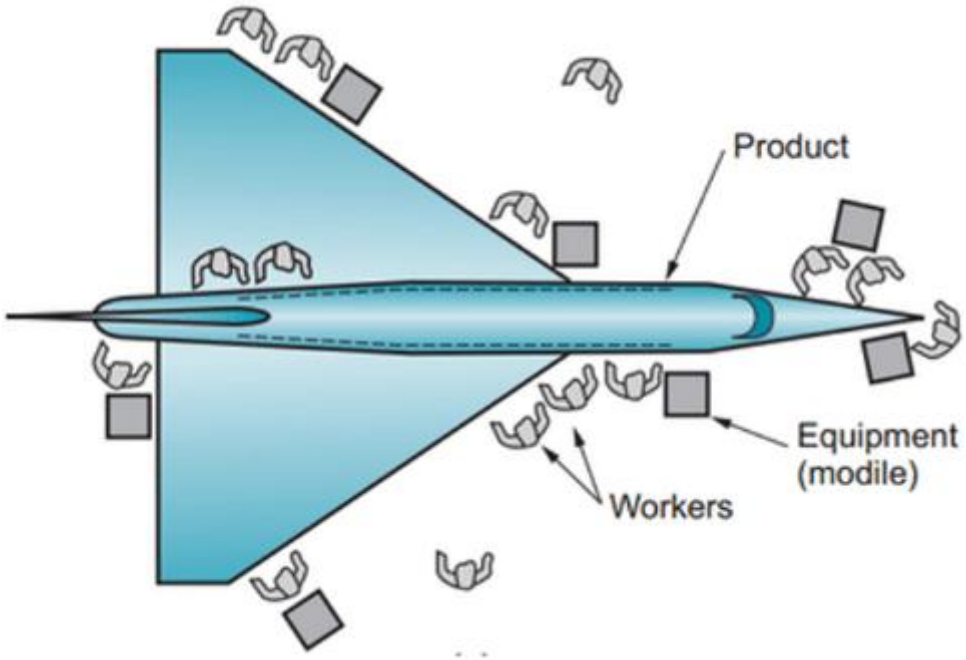
Tableau 1 : exemple d'un cheminement dans un job shop

Job Ji	**J1**			**J2**				**J3**	
Opération $O_{i,j}$	$O_{1,1}$	$O_{2,1}$	$O_{3,1}$	$O_{1,2}$	$O_{2,2}$	$O_{2,3}$	$O_{2,3}$	$O_{3,1}$	$O_{3,2}$
Machine $M_{i,j}$	M1	M2	M3	M2	M4	M3	M1	M2	M4
$P_{i,j}$	5	2	4	1	2	6	2	4	4

Dans l'exemple qui figurant dans le tableau ci-dessus, nous avons un atelier te type Job Shop disposant de quatre machines, trois tâches se font dans cet atelier, tel que le cheminement de la première tâche est comme suite : machine 1→ machine 2→ machine 3 pour des temps de traitement de 5, 2 et 4 unités de temps (respectivement) ; le cheminement de la deuxième tâche est : machine 2→ machine 4→ machine 3→ machine 1 pour des temps de traitement de 1, 2, 6 et 2 unités de temps (respectivement) ; et pour la dernière tâche : machine 2→ machine 4 pour des opérations qui durent 4 unités de temps chacune.

6.2.6 Cas d'un Project shop (fixed-position layout) : le processus produit est dans une position stationnaire fixe et que les ressources telles que les personnes, l'équipement et les machines vont au produit. Une position fixe est idéale pour les produits volumineux, lourds ou fragiles qui sont trop difficiles à déplacer. **[17] [18] [13]**

Figure 11 : Exemple d'agencement d'un processus à position fixe [17].



6.3 Le champ β (caractéristiques et contraintes) :

Le champ β exprime des restrictions sur les valeurs que peuvent prendre conjointement les variables de décision. Et donc, lors de la construction d'un ordonnancement, le respect des caractéristiques et contraintes exprimées dans ce champ est une condition sine qua non pour sa réalisation **[19]**. En se référant à [7] et [20], nous allons innumérer quelques contraintes et caractéristiques fréquentes dans les probèlemes d'ordonnancement :

6.3.1 La disponibilité (r_j) : normalement le tâches peuvent être traitées dans l'nterval de temps $[\mathbf{r_j} ; +\infty[$, mais lorsqu'une date limite est définie, la tâche doit être traitée dans l'interval $[\mathbf{r_j} ; \mathbf{d_j}]$. En cas d'occurrence, les retards seront taxés.

6.3.2 Préemption (prmp) : cette caractéristique permet l'interruption du traitement d'une tâche pour traiter une autre tâche, et reprendre le traitement interrompu plus tard (tâche exécutée par morceau). Dans le cas où la tâche peut reprise sur une autre machine, nous parlons de migration alors.

6.3.3 La précédence (prec) : cette contrainte oblige certaine tâches à passer seulement après le traitement d'autres tâches, en guise de motivation, nous donnons l'exemple d'une limonaderie, tel que l'opération de bouchonnage doit attendre celle du remplissage impérativement.

6.3.4 Le parallélisme : caractérise la possibilité de traiter une tâche par plusieurs machines simultanément.

6.3.5 Les pannes (brkdwn) : dans la réalité, cette contrainte est toujours présente, elle stipule que les ressources ne sont pas disponibles en permanence.

6.3.6 La permutation (prmu) : c'est la contrainte qui caractérise les flow shop de permutation (l'ordre de traitement des tâche reste le même pour toutes les machines).

6.3.7 Le blocage (block) : impossibilité d'une tâche de quitter une ressource, ce qui va causer un retard sur le plannig de production et qui veut dire par conséquent des frais supplémentaires.

6.3.8 Pas d'attente (no-wait, nwt) : le temps d'attente d'une tâche entre deux opération est toujours nul.

6.3.9 Recirculation (recrc) : dans l'environnement Job shop, les tâches passer plus d'une fois par une même machine.

6.3.10 Temps de traitement dépendant de la séquence (s_{jk}) : aussi appelé "effet de détérioration", le temps opératoire d'une tâche vari selon son ordre dans la séquence, tel que plus son traitement tarde pour commencer, le p_i croît linéairement avec lui (éxemple du recuit de fer, plus le recuit tarde, plus le fer refoidi et l'opération devien plus difficile, et donc prend plus de temps.)

6.3.11 Restriction sur les machines (M_j) : dans le cas d'un environnement de machines parallèles à machines différente (Rm), certaines tâches ne peuvent pas être traitées sur n'importequelle machine.

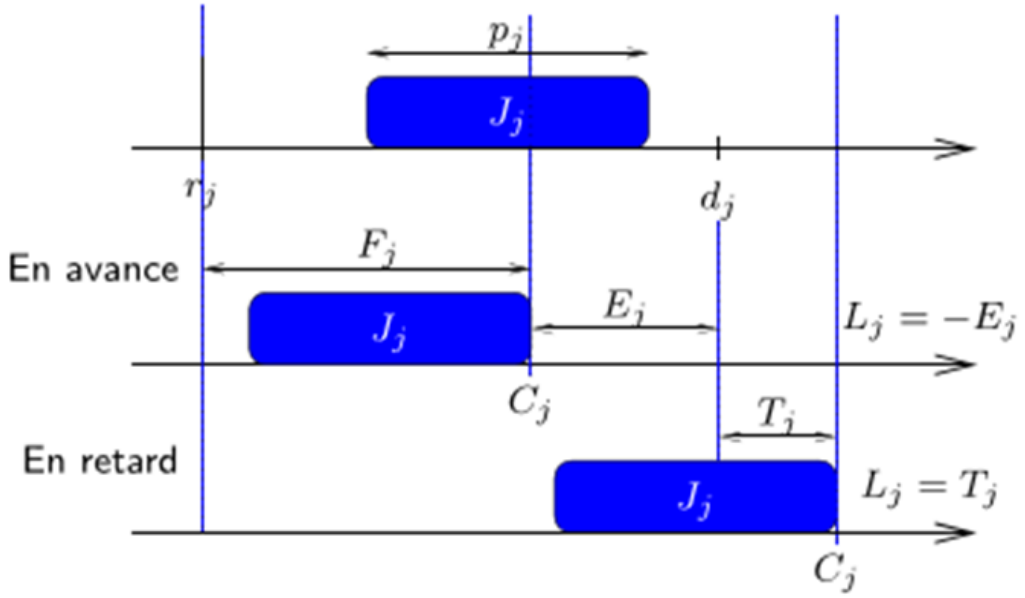


Figure 12 : exemple d'un problème d'ordonnancement sous une contrainte de disponibilité avec retard. [7]

6.4 Le champ γ (la fonction objectif) :

Comme dans chaque problème, nous cherchons à trouver une solution optimale (ou réalisable) pour une fonction donnée (cas d'un problème monocritère), ou trouver le meilleur compromis pour un ensemble de fonctions (cas d'un problème multicritères). Les objectifs à optimiser (aussi dits critères d'évaluation [19] ou encore critère d'optimalité [20]) sont des indicateurs de performance clés (**KPIs** pour Key Performance Indicators en anglais) sur lesquels se base une décision d'ordonnancement satisfaisante, généralement ces objectifs sont liés au temps, aux ressources ou encore aux coûts. Nous allons donner quelques exemples des plus fréquents dans la littérature des problèmes d'ordonnancement :

6.4.1 La somme des dates de fin ($\sum C_j$) : appelée **total completion time**, elle représente la somme des dates de fin de séjour (notée $\mathbf{C_j}$) de toutes les taches (date de sortie du système).

6.4.2 La somme pondérée des dates de fin ($\sum W_j C_j$) : appelée **total weighted completion time**, cet objectif donne des poids ($\mathbf{w_j}$) d'importance aux temps de séjour des tâches, cet objectif est pris en considération lorsque les coûts du temps de séjour ne sont pas égaux alors on essai de minimiser ce coût.

6.4.3 La durée totale de l'ordonnancement (C_{max}) : dite " **makespan** ", elle représente la date de fin du traitement de toutes les tâches, autrement dit, la valeur du makespan représente la plus grande valeur des dates de fin de traitement de toutes les tâches. $\mathbf{C_{max} = \max \{C_j\}}$.

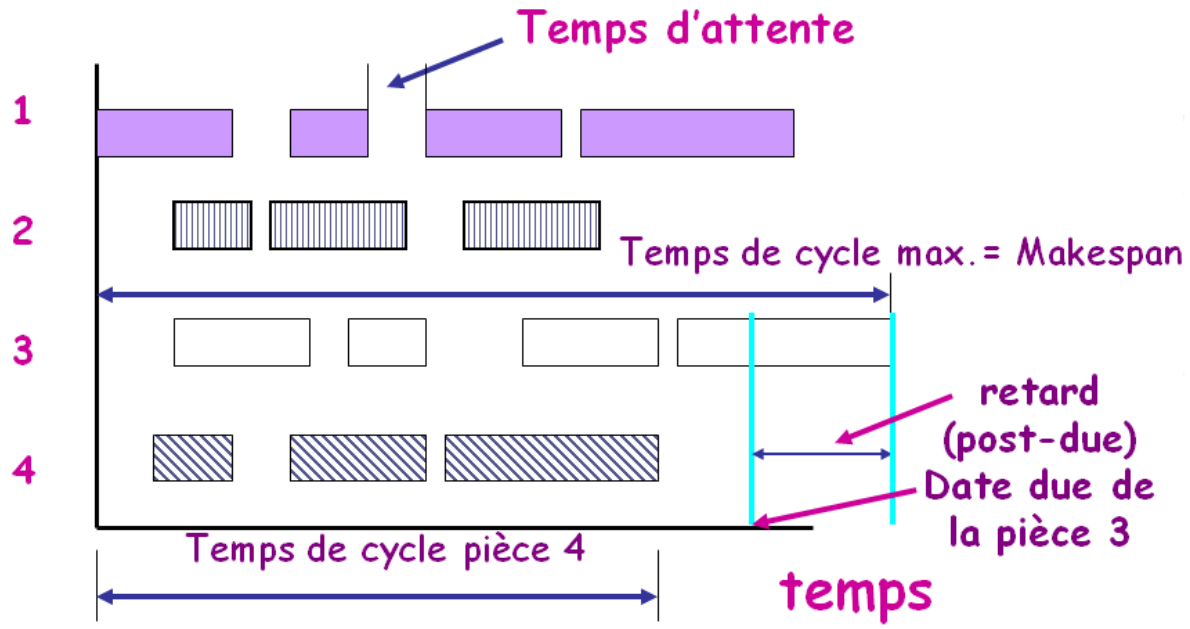
6.4.4 Le retard algébrique maximum (L_{max}) : le **maximum lateness** consiste à minimiser le plus grand retard, sa formule est donnée par $\mathbf{L_{max} = max \{L_j\}}$ tel que $\mathbf{L_j}$ est le retard algébrique et il est égal à $\mathbf{L_j = C_j - d_j}$.

6.4.5 La somme pondérée retard maximum ($\sum W_j T_j$) : appelée " **total weighted tardiness** ", tel que L_j est el retard absolu et il est donné par :

$$\mathbf{L_j = max \{0 ; C_j - d_j\}.}$$

6.4.6 Le nombre pondéré des tâches en retard ($\sum W_j U_j$) : le weighted number of tardy jobs permet de calculer le coût lié au nombre de tâches en retard, tel que $\mathbf{U_j}$ est la pénalité unitaire du retard, il est égal à 0 si ($\mathbf{C_j < d_j}$), 1 si non ; et le poids dans ce cas représente le coût du retard de chaque tâche.

Figure 13 : représentation du makespan, temps de cycle et du retard dans un diagrame de gantt. [7]



Alors pour trouver le bon ordonnancement de tâches, le plus souvent du temps, et en vue de la complexité des problèmes, des méthodes de résolution comme les méta-heuristique, ou encore l'apprentissage automatique et les multi-agents font l'affaire, mais quand le problème est plus basique, d'autres règles de passages sont utilisées :

6.5 Règles de passage :

Une règle de passage ou de priorité nous permet de déterminer un ordre dont le traitement des tâches doit suivre. Nous citons ci-dessous quelques exemples avec explication de leur fonctionnement (voir **[7]** et **[20]**) :

6.5.1 Shortest (expected) porcessing time " SPT ou SEPT " : cette règle stipule que le traitement des tâches se fait selon l'ordre croissant de leur temps de traitement. Il existe plusieurs variantes du SPT :

- TSPT (Truncated SPT) : la tâche ayant le plus court temps de traitement est choisie en 1er, mais s'il y a une tâche ayant un temps d'attente à **ω**, alors elle doit passer en 1er. (**ω** est une valeur choisi arbitrairement).
- SRPT (shortest remaining processing time).
- WSPT (weighted shortest processing time) : utilisée dans le cas d'un problème 1| |$\sum w_j C_j$, les tâches sont traitées dans l'ordre des $\frac{pi}{wi}$ croissants.

6.5.2 Longest (expected) processing time " LPT ou LEPT " : cette règle stipule que le traitement des tâche se fait selon l'ordre décroissant de leur temps de traitement. Elle est utilisée pour le problème de makespan par exemple : cette règle donne une solution satisfaisante pour le problème Pm| |C_{max}. Ceci veut dire que donné une solution intuitive optimale notée C_{max} (opt) nous trouvons :

$$\frac{\mathbf{Cmax\ (LPT)}}{\mathbf{Cmax\ (opt)}} \leq \frac{4}{3} - \frac{1}{3m}$$

6.5.3 Earliest due date first " EDD " : utilisée dans le cas d'un problème 1| |L_{max}, elle consiste à ordonner les tâches selon un ordre croissant de leur date échues.

6.5.4 Règle de Johnson " SPT(1) – LPT(2) " : utilisée pour les problèmes de type F2| |C_{max}, elle consiste à partitionner les tâches en deux groupes, et puis ordonnancer les tâches du 1er groupe en premier suivant la règle SPT, ensuite ordonnancer les tâches du 2ème groupe suivant la règle LPT. La répartition des tâches sur les deux groupes se fait comme suite : affecter les tâches dont le temps du traitement sur la première machine est plus petit que celui de la deuxième machine au premier groupe, et les tâches qui restent sont affecté au deuxième groupe.

6.6 Les méthodes de résolution :

Il est vrai que le nombre des méthodes de résolutions des problèmes d'ordonnancement ne cesse de croitre avec toutes les recherches académiques soient-elles ou du terrain, et donc le choix de " la méthode " est distinct pour chaque problème au dépend des besoins de l'ordonnanceur et les caractéristiques du problème [7], mais il est possible de les grouper dans deux classes :

6.6.1 Les méthodes exactes : ce sont des méthodes d'énumération qui permettent de trouver une solution optimale, par contre, ces méthodes ne peuvent pas résoudre tout type de problème, et il s'avère qu'elles sont très gourmandes en terme de temps d'exécution quand il s'agit de problème avec une taille considérable ou d'une complexité. Nous citons certaines méthodes :

6.6.1.1 Le branch and bound (la séparation et évaluation) : décomposition d'un problème initial en sous-problèmes pour former une partition de l'espace des solutions et évaluer les bornes optimales des solutions de chaque sous-problème.

6.6.1.2 La programmation dynamique : basée sur le principe d'optimalité de Bellman, cette méthode consiste en la résolution des sous-problèmes d'un problème global pour trouver la solution optimale.

6.6.1.3 La programmation linéaire : après que Leonid Kantorovič [22] ait proposé la technique mathématique que nous connaissons aujourd'hui par la programmation linéaire, c'est un mathématicien américain du nom de George Bernard Dantzig qui posa le problème de la programmation linéaire pour la première fois en 1947, qu'il a pu résoudre en introduisant la méthode simplex. La programmation linéaire consiste à minimiser une fonction linéaire avec des contraintes linéaires.

6.6.2 Les méthodes approchées : ce sont des méthodes itératives qui permettent de trouver une solution optimale dans le meilleur cas, ou de se rapprocher de la solution optimale dans le cas général, elle est utilisée dans la recherche des solutions satisfaisantes dans le cas où Les méthodes exactes :

- Ne peuvent pas trouver la solution optimale dans un lapse de temps raisonnable.
- Ne peuvent pas résoudre ce type de problèmes.

Nous distinguons deux types d'algorithmes de méthodes approchées :

6.6.2.1 Les heuristiques : ce sont des algorithmes dédiés à des problèmes donnés. On peut donner l'exemple des algorithmes glouton (algorithme qui fait le meilleur choix sur le moment sans retour en arrière ni anticipation des étapes suivantes).

6.6.2.2 Les méta-heuristiques : l'ajout par rapport aux heuristiques, est que les méta-heuristiques peuvent être adaptées et appliquées sur n'importe-quel problème. On peut citer les algorithmes génétiques (**AG**s) à titre d'exemple (algorithmes d'optimisation stochastique fondés sur les mécanismes de la sélection naturelle et de la génétique).

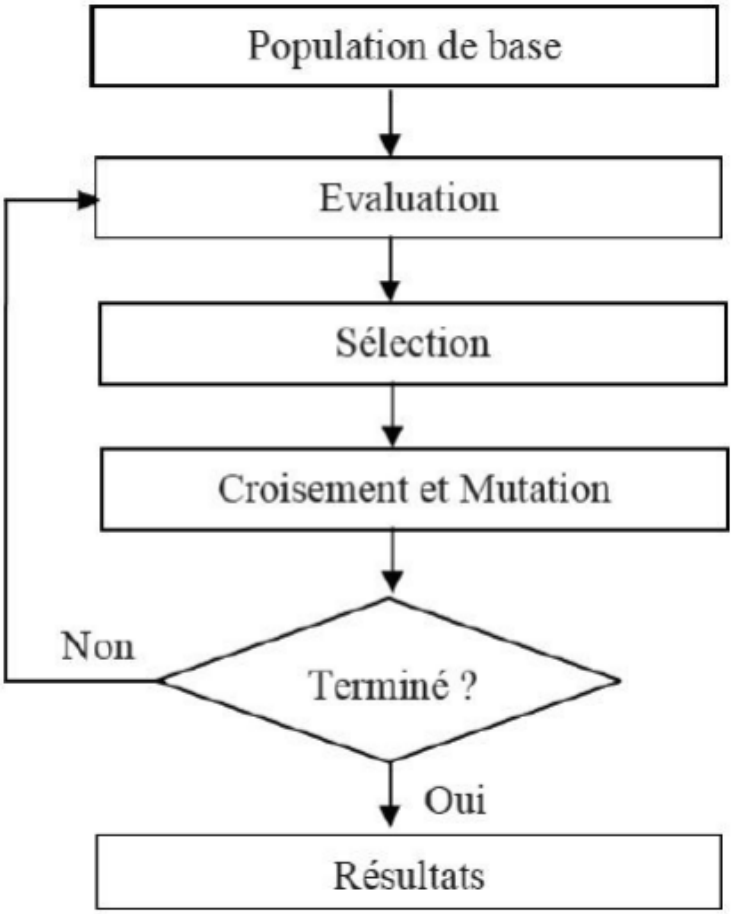


Figure 14 : Fonctionnement d'un AG.

I.7. Conclusion :

Dans ce chapitre, nous avons parlé l'importance d'avoir une bonne gestion de la production, et l'apport de l'ordonnancement pour la gestion de la production notamment. En outre, nous avons exposés les différentes notions dont on aura besoin pour comprendre la suite du travail, à savoir, les types d'ateliers qu'on trouver dans les industries, les types de problèmes d'ordonnancement ainsi que les méthodes de résolution.

Chapitre II : état de l’art.

Chapitre 2

II.1. Introduction :

On dit souvent que la science est un processus cumulatif, les membres de la communauté des scientifiques publient les résultats de leurs recherches pour partager avec ceux qui sont intéressés, ce qui permet un meilleur avancement dans les travaux de recherches. Ceci dit, afin de s'instruire, de s'illuminer, de s'inspirer et de s'informer des nouveautés du monde scientifique en guise de rester à jour, tout chercheur se doit de faire une recherche bibliographique, ça permet aussi de répondre à d'éventuelles question et de développer de nouvelles idées, ainsi, un examen structuré de la littérature par le biais d'une recherche bibliographique est nécessaire pour crédibiliser le travail d'un chercheur. De ce fait, nous avons consacré ce chapitre à la recherche bibliographique, nous allons faire un état de l'art sur l'ordonnancement en premier lieu, en outre nous allons retracer certains travaux des plus pertinents qui ont abordés les problèmes d'ordonnancement avec effet de détérioration pris en considération. Et pour finir, nous allons parler de la valeur ajoutée de notre travail sur les autres travaux qui l'ont précédé.

II.2. Problèmes d'ordonnancement :

2.1 Machines parallèles : on trouve dans sa littérature que les problèmes d'ordonnancement sur un atelier à machines parallèles sans préemption sont généralement des problèmes NP-hard, en l'occurrence $R||C_{max}$ et $P||C_{max}$; pour le premier problème, un algorithme branch and bound serait un bon choix, mais pour le deuxième problème, nombreuses heuristiques ont été adoptées afin d'approcher le plus la solution optimale. [20]

On a proposé dans [24] pour le problème $R_m|setup\ time|\ C_{max}$ étant un problème NP-hard, une méta-heuristique pour la recherche aléatoire des priorités (meta-heuristic for randomized priority search) **meta-raps**, et on a également fait une modélisation mathématique en nombres entiers mixtes. Plus tard le même problème a été traité dans [26] où l'on a proposé un algorithme de colonie de fourmis à deux phases, plus performant que le précédent, cet article était la continuité du travail fait dans [25] tel qu'on a proposé des améliorations et expérimentations.

Cependant, la considération de la préemption permet de réduire la difficulté du problème, à titre d'exemple, la procédure McNaughton permet de résoudre en un temps O(n) le problème $P|pmnt|C_{max}$, et ce de façon optimal. Pour les problèmes avec contrainte de précédence d'arborescence $1|p_i = 1, tree|C_{max}$; la règle CP nous donne de bons résutats dans le cas où l'arborescence est un arbre descendant ou ascendant.

Quant aux machines uniformes, la règle LRPT peut être utilisée pour résoudre un problème de type $Q_m|pmnt|C_{max}$,en assignant la tâche à la machine disponible la plus rapide. Si l'on change la fonction objectif pour $\sum C_i$, on utilise la règle SRPT.

Le tableau suivant retrace certaines recherches portant sur la minimisation du Cmax dans un atelier où les machines sont parallèles.

Tableau 2 : problèmes de minimisation du makespan sur des machines parallèles.

Problème	Référence
$R_m\|\| C_{max}$	[27], [28], [29], [30], [31], [32]
$R_m\|r_i\| C_{max}$	[33]
R_m|ress contraint| C_{max}	[34]
$R_m\|d_i\| C_{max}$	[35]
$R_m\|st\| C_{max}$	[39], [26], [36], [37], [38]
$R_m\|\ d_i, r_i, st\| C_{max}$	[40]
$R_m\|r_i$, st, prec, ress$\| C_{max}$	[41]
R_m|batch$\| C_{max}$	[42]
R_m|ma$\| C_{max}$	[43]

2.2 Flow-shop hybride :

la combinaison des deux problèmes d'ordonnancement sur flow-shop et sur machines parallèles a donné naissance aux problèmes d'ordonnancement sur ateliers flow-shop hybride (aussi appelé flow-shop flexible) ; pour ce type de problème, la préemption n'est pas permise, et nous rappelons aussi que dans un niveau les machines peuvent être : (**i**) identiques, (**ii**) uniformes ou (**iii**) générales.

Le problème du flow-shop flexible a été initié par Arthanari et Ramamurthi [44], il a été prouvé que ce problème est NP-hard pour les critères d'optimisation classique [45].

la notation la plus utilisée dans pour ce problème est celle qu'avaient proposés Vignier et Billaut [45], et qui généralise celle de Graham [8] $\alpha|\beta|\gamma$ expliquée dans le premier chapitre, cette notation consiste à diviser α en quatre termes pour formuler le type d'atelier comme suite $\alpha_1\alpha_2(\alpha_3\alpha_4^{(1)}, \alpha_3\alpha_4^{(2)}, \alpha_3\alpha_4^{(3)}, \ldots)$, tel que : α_1 nous indique le problème traité (dans notre cas c'est HF qui désigne flow-shop hybride), tandis que α_2 nous informe sur le nombre de niveaux qui est forcément supérieur à 1, α_3 spécifie le type de machines par niveau (**P** pour identiques, **Q** pour uniformes et **R** pour générales ou **0** s'il n'a qu'une seule machine), et finalement pour indiquer leur nombre. Donnons un exemple pour mieux assimiler : la notation HF3($P3^1$, 0^2, $Q2^3$) signifie qu'il s'agit d'un système flow-shop flexible à 3 niveaux, le 1^{er} niveau contient 3 machines parallèles identiques, le $2^{ème}$ niveau contient une seule machine et le $3^{ème}$ et dernier niveau contient 2 machines parallèles uniformes.

Le tableau ci-dessous énumère certains travaux de recherches sur des problèmes d'ordonnancement dans un atelier flow-shop classique.

Tableau 3 : problèmes flow-shop classique.

Problème	Référence
C_{max}	[47], [48], [49], [50], [51]
$\sum C_i$	[52]
No wait, C_{max}	[53], [54]
No wait, $\sum C_i$	[55], [56]
Pas de stock intermédiaire, F	[57]
No wait, L_{max}	[59]
C_{max}, $\sum w_i C_i$	[58]

Le tableau suivant présente quelques travaux portant sur les problèmes de flow-shop hybride.

Tableau 4 : problèmes flow-shop hybride.

Problème	Référence
HF2(0,$P^{(2)}$)	
C_{max}	[60]
$\sum C_i$	[61]
HF2(0,$R^{(2)}$)	
C_{max}	[62], [63]
HF2(0,$Q^{(2)}$)	
C_{max}	[64]
HF2($P^{(1)}$, $P^{(2)}$)	
C_{max}	[65]
$C_{max} + \sum T_i$	[66]
HFk($P^{(1)}$, $P^{(2)}$, …, $P^{(k)}$)	
C_{max}	[67], [68]
$\sum w_i C_{max}$	[69]
C_{max}, F	[70]
$\sum C_i$	[71]
C_{max}, T_{max}	[72]

II.3. Effet de détérioration :

On a toujours considéré le temps de traitement fixe dans les problèmes d'ordonnancement classiques, cependant, il existe bel et bien dans la réalité, des modèles où le temps de traitement change selon certains facteurs. Il existe deux types de détérioration, (**i**) la détérioration est liée à la tâche (on parle dans ce cas de tâches à effet de détérioration, ou tâches déteriorables), donnons l'exemple du temps de traitement des laminoirs d'acier, tel que le temps requis pour sa fabrication augmente proportionnellement avec la baisse de la température ; (**ii**) la détérioration est liée à la ressource, où la ressource perd de sa cadence et de ses performances proportionnellement avec le temps ou son utilisation, donnons l'exemple d'un opérateur dont le rendement diminue lorsque le taux de fatigue augmente, lorsque la détérioration et liée à la ressource comme dans ce cas, nous utilisons des **RMA** (rate modifying activity) qui servent à augmenter le rendement de la ressource (pour l'exemple donné une pause de récupération est considérée comme RMA car la pause permettra à l'opérateur de récupérer et de se concentrer mieux).

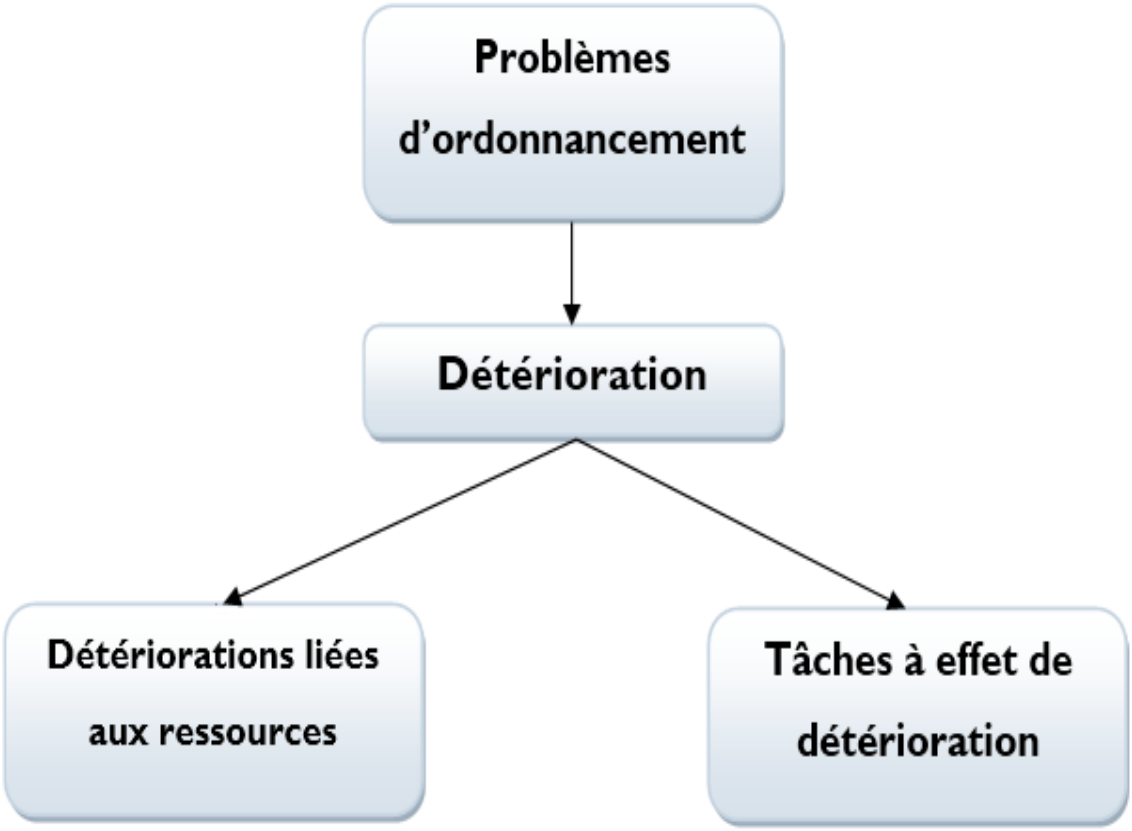


Figure 15 : **effet de détérioration.**

Chapitre 2

Dans ce qui suit, nous allons présenter un état de l'art des recherches effectuées, afin de retracer les travaux faits sur l'ordonnancement des tâches en prenant compte de l'effet de détérioration :

3.1 Tâches déteriorables : le concept des tâches déteriorables a été introduit par Gupta et Gupta [73] en 1988, ensuite et indépendamment par Brown et Yechiali [74] en 1990 dans les recherches précurseuses sur la détérioration, ils ont supposé que les temps de traitement des tâches varient selon une fonction de détérioration linéaire donnée par l'équation $p_i = p_i^0 + \alpha_i t_i$ (tel que p_i, p_i^0, α_i et t_i représentent respectivement le temps de traitement actuel de la tâche i, le temps d'exécution initial de la tâche i, le taux de détérioration ainsi que le temps où l'on a commencé le traitement de la tâche i), et ont appliqué la fonction pour le temps de traitement réel dans un problème d'ordonnancement sur une seule machine ayant pour critère d'optimisation le makespan.

Wang et al. dans [75] ont traité ce problème sur une seule machine ayant considéré des contraintes de précédence, et ont pu prouver qu'il existe un ordonnancement optimal pour minimiser le makespan lorsque la relation de précédence est sous forme de chaine, et pour quand elle est sous forme de graphe série parallèle, ils ont présenté un algorithme.

Pour les ateliers à machines parallèles, Kuo et Yang [76] ont traité le problème $Pm|\mathbf{p_i=p_{ij}^0+\alpha_{ij}t_{ij}}|F$ à l'aide d'un algorithme polynomial, Li et Yuan [77] ont proposé un schéma d'approximation en temps polynomial pour le problème $Pm|\mathbf{p_i=p_{ij}^0+\alpha_{ij}t_{ij}}|C_{max} + \sum e_i$. Tigane et Dahane [14] ont considérés dans cet article le problème d'ordonnancement des tâches détériorables sur des machines parallèles non liées. L'objectif étant de trouver le meilleur ordonnancement minimisant la consommation totale d'énergie (en guise de minimiser le taux d'émission du CO2) et le makespan, deux approches basées sur NSGA-II ont été présentées : (**i**) dans la première, le classement des tâches était basé sur la règle d'ordonnancement des tâches sur une seule machine ; dans ce cas, l'objectif était de déterminer la meilleure affectation des tâches pour chaque machine ; ensuite, une méthode exacte pour valider cette première approche a été mise en œuvre, (**ii**) dans la deuxième approche, l'objectif était de trouver la meilleure affectation des tâches et le meilleur ordre sur chaque machine. Et bien sûr comme les deux objectifs sont contradictoires, il fallait trouver le meilleur compromis en utilisant une analyse multicritère, la méthode **TOPSIS** a été utilisée pour trouver la solution voulue, et pour finir, une approche améliorée a été proposée, l'algorithme NSGA-II amélioré permet de trouver le front de Pareto optimal, et les résultats numériques ont bel et bien prouvé que cette approche améliorée fournit de meilleur résultats. Dans sa thèse de doctorat [20], Tigane a retravaillé sur ce même problème, et par la suite elle a utilisé la même approche pour ce même problème dans un flow-shop hybride, un autre algorithme a été implémenté pour comparer entre les deux, il s'agit de de l'algorithme MOMSA introduit par Lin et Ying [78], et qui est une méta-heuristique pseudo front-pareto qui se base sur l'algorithme du recuit simulé, les

résultats montrent que l'algorithmes génétique est quasi-majoritairement plus performant pour les petites et grande instances.

Li et al. [79] ont fait une extension du problème du flow shop à permutation distribuée (DPFSP) avec des contraintes d'ordre. Tel qu'on a considéré l'ordonnancement de tâches à effet de détérioration sur 3 usines identique munies d'un moyen de manutention entre les machine (un seul robot par usine), pour résoudre ce problème, un algorithme iterative greedy amélioré (**IIG**) qui présente la contribution suivante : (1) chaque solution a été représentée par un vecteur à deux dimensions, où le premier vecteur représente l'affectation de l'usine et le second l'ordonnancement des tâches ; (2) en tenant compte de l'acheminement du robot pendant le processus de transport, une heuristique de décodage efficace est développée ; (3) pour améliorer les calculs étant complexes, une approche efficace de destruction et de construction a été intégrée à l'algorithme proposé (cette phase est sert à générer des solutions à chaque itération) ; (4) pour équilibrer les capacités de recherche globales et locales, quatre opérateurs de recherche de voisinage spécifiques au problème ont été présentés ; et (5) le critère d'acceptation SA-based a été intégré pour améliorer les capacités de recherche globale (à chaque itération on re-calcul la probabilité d'évasion et la température), pour démontrer l'efficacité du critère d'acceptation SA-based, une comparaison avec un **IIG-NS** (sans SA-based) a été faite, et les résultats on montrés que cette méthode a amélioré le processus de recherche globale, ensuite, une autre comparaison a démontré que **IIG** est plus performant qu'un algorithme sans recherche locale (**IIG-NL**), et enfin, on a comparé cet algorithme avec d'autres approches utilisées dans d'autres travaux similaires, en l'occurrence : l'algorithme ABC-L [80], L'algorithme IG-R [81], l'algorithme ICA-L [82], et il été claire que l'IIG été plus performant sur le grandes et petites instances.

Fiszman et Mosheiov [103] ont traité le cas d'un problème de flow-shop proportionné ayant pour critère d'optimalité le total load (minimiser la somme des makespan sur l'ensemble des ressources du système), c'est un objectif rarement traité dans les recherches, mais il est justifié lorsque le coût est exprimé en fonction du temps où les machines sont occupées, les responsable de la planification et de l'ordonnancement essaient dans ce cas de terminer le travail sur les machines le plus tôt possible. Ici, on n'a pas considéré la détérioration classique de la ressource, mais on a supposé que le temps de traitement d'une tâche est défini au préalable et au dépend de sa position dans l'ordre de traitement (on parle alors position dependent deterioration). Dans le but de trouver le meilleur ordonnancement de n-tâches sur m-machines. Fiszman et Mosheiov ont pris en considération l'effet de détérioration dépendant de la position des tâches, autrement dit, l'ordre de chacune des tâches dans leur traitement définit le temps de traitement requis pour celles-ci, donc ils assument que qu'il n'y ait pas de motonicité du temps de traitement (le temps de traitement est soit croissant ou décroissant). Une extension de ce problème a été traitée par la suite qui consiste à prendre en considération le job-rejection, ce qui nous fait deux objectifs linéairement dépendant à minimiser. En effet le job-rejection est devenu durant la décennie une tendance dans le domaine de la recherche concernant les problèmes d'ordonnancement, nous faisons face à ce problème lorsque nous devons éviter de retarder une tâche importante (sur le plan financier ou autre). Au tout début, Fiszman et Mosheiov ont fait un modèle mathématique résoluble pour O(n5) time, et O(n6) time pour l'extension du problème avec job-rejection. En utilisant un algorithme d'amélioration publié dans l'article d'Agnetis et Mosheiov [104], on a amélioré le temps

d'exécution des deux algorithmes avec un facteur de n, ce qui donne O(n4) time et O(n5) time.

Wang et Ji [105] ont démontré que la méthode classique d'ordonnancement sur deux machines en série nommée la loi de Johnson (voir chapitre 1), ne donne pas forcément la solution optimale pour le problème $F2|p_{i,j,r}|Cmax$. En outre, on a trouvé des solutions en temps polynomiales meilleures pour des problèmes spécifiques ayant pour fonctions objectifs : le weighted sum of completion times et le maximum lateness. On a considéré dans tous ces problèmes traités que le temps de traitement dépend de la position de la tâche (position dependent) dans l'ordre de traitement. Une extension du problème a été discutée en mentionnant des travaux concernant l'apprentissage et non la détérioration en l'occurrence [106] qui a présenté une étude informatique des différents modèles de courbe d'apprentissage, et on a fait une extrapolation avec d'autres études pour intégrer la dépendance du temps de la position d'une tâche.

Note : généralement quand on parle de time dépendent avec effet de détérioration nous somme dans le cas du temps de traitement croissant, autrement si le temps est décroissant, nous parlons alors d'apprentissage.

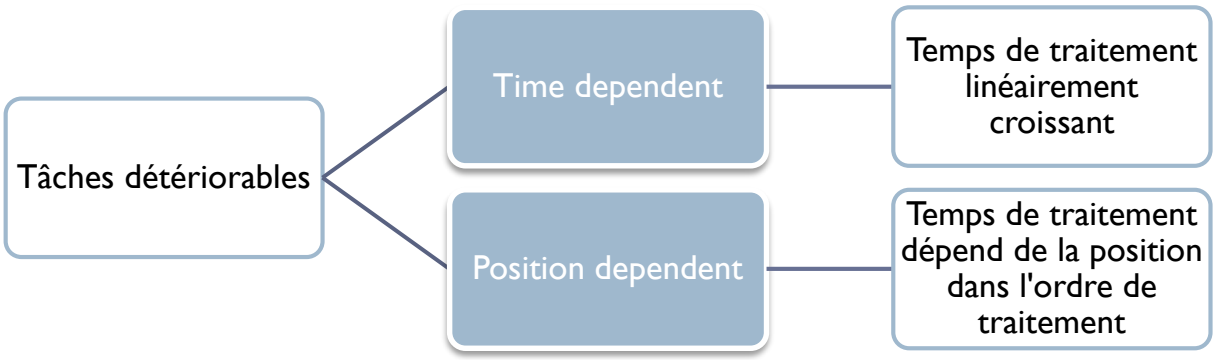


Figure 16 : résumé des tâches à effet de détérioration.

3.2 Détérioration liée à la ressource : bien que les problèmes d'ordonnancement où le temps de traitement dépend du taux de production de la ressource et qui se dégrade sont les plus traités comparés à ceux dont on considère les tâches détériorables, mais le concept du RMA a été initié par Lee et Leon [83] seulement en 2001, ils stipulent qu'une RMA est une activité qui modifie le taux de production de la ressource. Par conséquent, les temps de traitement des tâches varient selon si le travail est programmé avant ou après l'activité notre RMA. Aussi loin qu'on ait cherché dans la littérature, nous avons trouvé que les travaux portant sur l'ordonnancement ou le séquencement prenant en compte l'effet de détérioration et le RMA peuvent être classées en trois catégorie :

- Problèmes d'ordonnancement avec détériorations dépendant du temps (time dependant problem), autrement dit, trouver le meilleur ordonnancement pour que l'effet de détérioration lié au temps affecte le makespan le minimum possible (voir [85], [86], [20], [87], [88], [89])
- Problèmes d'ordonnancement avec RMA (position dependent problem), et ce, trouver le nombre et la position optimale des RMA (voir [84], [90], [91], [92])
- Problèmes d'ordonnancement avec détériorations dependent du temps et RMA (time and position dependant problem), traiter les deux contraintes simultanément (voir [84], [79], [93], [94], [95], [96], [97], [98], [99], [100], [101], [102]).

Woo et Jung [84] ont cherché à trouver le meilleur ordonnancement des tâches sur des machines parallèles avec des temps de traitements différents, mais aussi en prenant compte la détérioration lié au temps (time dependant deterioration : plus une tâche tarde pour être traitée l'effet de détérioration augmente et le temps de traitement proportionnellement), ainsi que l'activité modificatrice des taux (RMA) en assumant qu'aucune tâche ne peut être traité durant une RMA, pour traiter le problème, 4 méta-heuristiques génétiques ont été développé, et comparées pour déterminer l'algorithme le plus performant, Les algorithmes sont (i) GA with triple-dimensional string (GA-TS), (ii) GA with completion time ruled-based dispatching heuristic (GA-RD), (iii) et (iiii) sont deux algorithmes dérivés des deux premiers respectivement grâce à un algorithme d'amélioration (GA-TSI) et (GA-RDI) respectivement (l'heuristique d'amélioration en question choisit de façon itérative les tâches dans un ordre décroissant des temps de traitement actuels, et affecte la tâche avec le temps de traitement le plus grand à la machine qui peut finir son traitement avant les autres en considérant une éventuelle application d'RMA si le temps requis pour la RMA est plus petit que le temps ajouté faute de la détérioration).

Lodree et al. [102] se sont intéressé à un problème de séquencement de tâches en prenant compte des deux classes de problèmes d'ordonnancement cités en haut, considérant une seule ressource qui est l'opérateur, on est parti du principe qu'une seule RMA peut se faire et ce en guise de minimiser le makespan. La contribution de cet article consiste à trouver la meilleure position de la RMA pour minimiser le C_{max}. On a trouvé en se basant sur un ensemble de lemmes et d'axiomes que la position optimale (k*) est donnée pour $\boldsymbol{k = \frac{n}{2} + 1}$ avec n représente le nombre de tâche à séquencer. Le choix de faire ou ne pas faire une RMA se fait après comparaison des deux makespan avec et sans RMA la plus petite valeur est retenue.

Sekkal et Belkaid [107] ont étudié et modélisé le comportement d'un problème d'ordonnancement de machines parallèles par rapport au temps de traitement. La machine est sujette à la détérioration et comprend deux contraintes de consommation de ressources. La première ressource R1 contrôle le temps de traitement de sorte que des quantités additionnelles de R1 diminuent le temps de traitement. La seconde ressource R2 est contrôlée par le temps de traitement réel, de sorte que la consommation de R2 est une fonction linéaire du temps de traitement réel. L'augmentation de la consommation de R1 entraîne un temps de traitement et donc une diminution de R2. La résolution de ce problème consiste à trouver l'ordonnancement optimal qui minimisera à la fois le makespan et le coût des ressources. Dans cet article, on a introduit un recuit simulé multi objectif (MOSA) afin de résoudre le problème d'optimisation combinatoire lié à la recherche de la meilleure combinaison (machine, travail, position, R1). Il s'agit littéralement de la meilleure affectation des tâches et de l'allocation des ressources afin de minimiser le makespan et le coût des ressources. Afin d'améliorer la qualité des résultats, on a également développé un algorithme en deux étapes en décomposant le problème original en deux sous-problèmes : un problème d'affectation et un problème d'allocation de ressources. Des simulations ont été réalisées pour analyser les performances des deux algorithmes. Les résultats ont montré que l'algorithme à 2 étapes est très efficace et surpasse MOSA

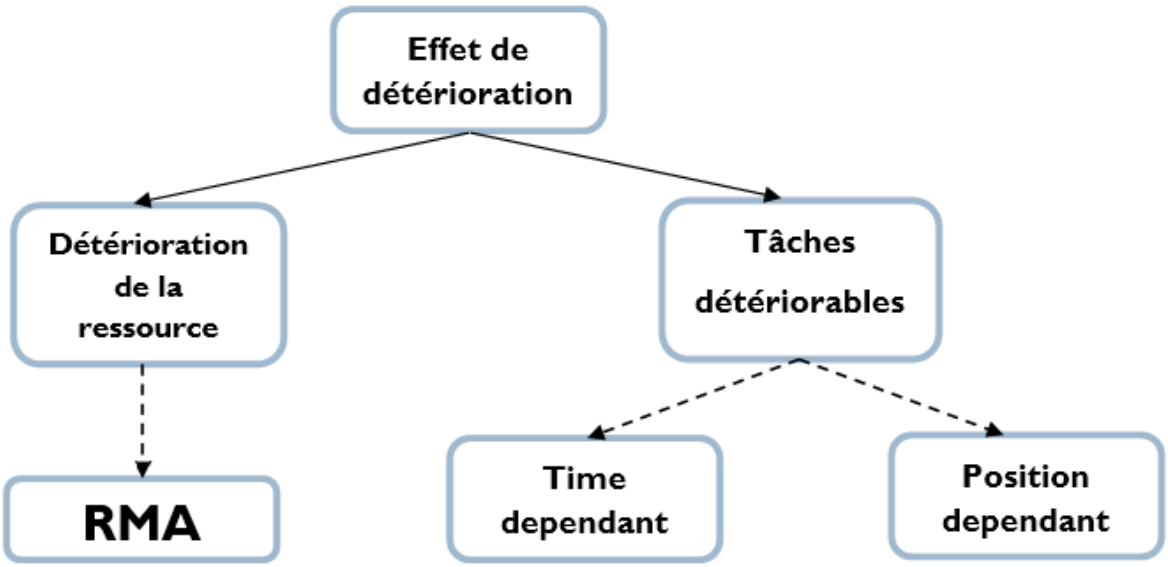


Figure 17 : **résumé des types de détériorations dans les problèmes d'ordonnancement.**

II.4. La valeur ajoutée de notre travail :

Aussi loin que nous sommes allés dans nos recherches, aucune étude ne s'est intéressée à considérer l'effet de détérioration dans un problème de tournées de véhicules (**VRP**), notre approche consiste à appliquer développer un algorithme génétique ayant pour objectif de minimiser le temps de livraison des produits dans une zone urbaine, où l'on va considérer le temps de livraison variable (détériorable), l'augmentation du temps de livraison peut être justifié par un éventuel encombrement de la circulation automobile, par la dégradation du véhicule suite à son utilisation

(chauffage du moteur en l'occurrence) qui requière de un arrêt éventuel puis une vigilance lors de la conduite (vitesse réduite), ou enfin la fatigue du chauffeur dont régi une baisse de la vitesse de roulement ou un éventuel arrêt.

II.5. Conclusion :

Dans ce chapitre nous avons fait un état de l'art des problème d'ordonnancement, puis on a détaillé pour les problèmes d'ordonnancement prenant en compte l'effet de détérioration, tel qu'il existe deux type de détériorations : détérioration de la tâche et détérioration de la ressource. En fin, ayant considéré le problème comme un problème d'ordonnancement, nous avons parlé de l'ajout de ce travail sur les travaux précédents, tel que ce travail est le premier à considérer l'effet de détérioration sur un problème de tournées de véhicules.

Chapitre III : application de la détérioration sur un VRP.

Chapitre 3

III.1. Introduction :

Etant donné l'importance du problème de tournées de véhicules, et l'intérêt que portent les entreprises au développement de modèles permettant d'optimiser la tournée que suit sa flotte, voir même réduire le nombre de véhicules nécessaire pour subvenir aux besoins de ses clients, nous nous sommes intéressés au problème de VRP, tout en considérant la contrainte de détérioration (voir le chapitre 2) et qui engendre l'accroissement du temps requis pour effectuer une livraison. Dans notre travail, nous allons travailler sur les données d'une entreprise de livraison des sachets de lait, tel que nous allons considérer la fatigue du chauffeur allant engendrer une diminution de la vitesse du véhicule et donc une augmentation dans le temps de livraison est une nouvelle approche de ce problème. Pour avoir des résultats fiables, nous proposons d'implémenter trois méta-heuristiques et comparer entre elles, nous citons : (**i**) un algorithme génétique, (**ii**) une méta-heuristique de recuit simulé et (**iii**) une hybridation des deux algorithmes précédents.

Nous allons procéder à la résolution du problème en trois étapes : résolution d'un problème VRP classique de 21 points de livraison et un entrepôt, puis intégrer un modèle de détérioration linéaire pour ce même problème, et pour finir, nous allons chercher le moment opportun pour appliquer notre RMA.

III.2. Vehicle Routing Problem (VRP) :

En français problème de la tournée de véhicules, le VRP étant classé parmi les problèmes de la recherche opérationnelle et de l'optimisation combinatoire, il consiste à modéliser une situation où une flotte de véhicules devrait livrer à des clients, ou de collecter de chez des fournisseurs notamment, le véhicule peut être un camion si l'on parle de transport, un robot de manutention si l'on appliquait ce problème dans une usine, une ressource humaine faisant une tournée d'intervention (maintenance, réparation et contrôle…), etc. l'objectif d'un VRP peut être la minimisation du temps d'utilisation des véhicules, la minimisation de la somme des distances parcourues par les véhicules ou la minimisation de la plus grande distance parcourue par l'un des véhicules, tout comme ça peut être le coût de livraison des biens. Le VRP est une extension du TSP (problème du voyageur de commerce ou aussi le problème du dernier kilomètre) qui prend en considération un seul véhicule.

III.3. La logistique urbaine :

Désignant l'ensemble d'actions qui visent à assurer l'approvisionnement des agglomérations (concerne les flux, l'acheminement et la livraison de marchandise et de biens dans la ville), bien qu'elle soit mal vue par le grand publique, elle demeura d'une priorité inébranlable en ce qui concerne l'assurance des vivres dans les agglomérations. Certes, les véhicules utilisés sont source d'encombrement d'espace, de pollution et

nuisance sonore, mais ça reste le meilleur moyen pour subvenir aux besoins des citadins qui ne cesse de croitre. Et c'est exactement là où consiste le défi, trouver des solutions pour réduire les facteurs désagréables pour la population, à titre d'exemple, réduire le nombre de véhicules circulants tout en gardant la même efficience, trouver les périodes opportunes pour livrer les commandes sans déranger les habitants…

III.4. Formulation du problème :

Il est vrai que l'optimisation de la tournée des véhicules représente le sujet de nombreuse recherches, et ce, vu la complexité de ce problème d'une part, en plus des bénéfices qu'engendre un bon ordonnancement de ses véhicules en l'occurrence la diminution des coûts d'autre part (moins de kilomètres parcourus revient à dire moins de carburant consommé, et moins pannes et vérifications nécessaire, …), la livraison dans des délais plus courts ce qui s'avère très important avec l'accroissement de la compétitivité qui est déjà serrés. Cependant, on n'a jamais considéré une détérioration qui pourrait retarder la livraison. Dans ce problème que nous proposons, nous allons essayer de minimiser le temps de livraison, où l'on va considérer une détérioration affectant la vitesse moyenne dont roule le véhicule, en effet, en prenant en considération le cumul de la fatigue du chauffeur, il est justifié que la vitesse dont il va rouler diminuerait qu'en temps normal, et donc la durée requise pour effectuer une livraison serait plus grande, nous allons considérer par conséquent que la vitesse du véhicule est proportionnellement liée à la fatigue (détérioration ou l'état de fatigue) du chauffeur, et que cette détérioration est une fonction linéaire (voir chapitre 2).

Nous étudions le cas d'une usine de fabrication des sachets de lait de vaches qui livre quotidiennement du lait pour ses 21 clients, ayant une flotte de trois camions, elle cherche à livrer d'autres villes. Pour ce, et au lieu d'acheter plus de véhicules afin de subvenir aux besoins de ses clients, nous proposons une alternative moins couteuse et qui est l'optimisation des temps de livraison pour que les chauffeurs aient suffisamment de temps pour se déplacer vers les autres régions où l'on souhaite livrer. Donc, nous allons travailler sur un problème VRP en considérant la détérioration du chauffeur, et en cherchant le meilleur moment pour qu'il fasse une pause en guise de se reposer (réinitialisation de la détérioration à 0). Ci-dessous la matrice de distances entre les clients et l'entrepôt (l'entrepôt est représenté dans la dernière colonne et ligne)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	entrepôt
1	0	124.5	261.1	331.8	600.4	447.2	491.3	523.2	826.2	563	577.7	414.8	341.8	304.5	749.6	595	339.4	573	731.5	683.1	727.2	740.3
2	124.5	0	136.6	207.3	475.9	322.7	366.9	398.8	701.8	513	564.8	312.2	466.3	429	874.1	719.5	463.9	838.7	718.5	807.6	851.7	864.8
3	261.1	136.6	0	262.9	531.5	209.1	230.3	360.7	663.7	376.4	428.2	449.3	602.9	565.6	1010.7	856.1	600.5	702.1	581.9	895	988.3	1001.4
4	331.8	207.3	262.9	0	274	200.6	489.4	248.1	551.1	635.6	687.3	519.5	673.6	636.3	1081.4	926.8	671.2	961.2	841.1	1154.1	1059	1072.1
5	600.4	475.9	531.5	274	0	474.6	763.5	522.1	825.1	909.6	961.4	788.1	1045	904.9	1249.2	1247.2	939.8	1235.3	1115.1	1428.2	1489	1165.3
6	447.2	322.7	209.1	200.6	474.6	0	288.8	151.6	454.6	434.9	486.7	507.8	789	751.7	1196.8	1042.3	786.6	760.6	640.4	953.5	1446.8	1187.6
7	491.3	366.9	230.3	489.4	763.5	288.8	0	440.4	481.9	169.7	197.9	219	633.5	596.1	1041.3	886.7	631	471.8	351.6	664.7	1157.9	1032
8	523.2	398.8	360.7	248.1	522.1	151.6	440.4	0	303	586.6	638.3	659.5	865.1	827.7	1272.9	1118.3	862.6	912.2	792.1	1105.1	1250.5	1263.6
9	826.2	701.8	663.7	551.1	825.1	454.6	495.4	303	0	312.2	590.2	611.3	1025.7	988.4	1433.5	1278.9	1023.3	864.1	743.9	1057	1550.2	1424.3
10	563	460.5	348.7	607.8	881.9	407.2	183.2	558.8	312.2	0	277.9	299.1	713.5	676.2	1121.3	966.7	711.1	551.8	431.7	744.7	1238	1112
11	577.7	475.1	428.2	687.3	961.4	486.7	197.9	638.3	590.2	277.9	0	313.7	728.2	542.4	1007.9	799.1	642.1	273.9	153.7	466.8	960	1126.7
12	356.7	267.8	449.3	475.1	743.7	507.8	219	659.5	611.3	299.1	313.7	0	414.5	377.1	822.2	667.7	412	587.6	467.5	780.5	799.9	813
13	353.5	478	614.6	685.3	978.7	800.7	637.5	876.8	1029.7	717.5	732.2	418.5	0	303.1	522.1	593.6	338	571.6	886	681.7	725.8	398.5
14	279	403.4	540	610.8	879.4	726.2	562.9	802.2	955.2	643	542.4	343.9	336.7	0	587.9	408.2	177.7	268.5	691.7	378.6	535.2	735.3
15	560.1	684.6	821.2	891.9	907.1	1007.4	844.1	1083.4	1756.5	924.2	938.8	625.1	206.6	509.7	0	408.8	544.6	778.2	1092.6	888.3	581.8	326.9
16	557.1	681.6	818.2	888.9	1115.9	1004.3	841.1	1080.3	1233.3	921.1	799.1	622.1	415.4	420.9	208.8	0	255.7	525.2	938.7	625.6	173	535.7
17	313.9	438.3	574.9	645.7	914.2	761.1	597.8	837.1	990.1	677.9	692.6	378.8	371.6	177.7	464.5	255.7	0	368.2	791.3	478.3	387.8	791.3
18	547.5	671.9	702.1	961.2	1235.3	760.6	471.8	912.2	864.1	551.8	273.9	587.6	605.2	268.5	734	525.2	368.2	0	395.4	202.9	663.8	1003.8
19	731.5	628.9	581.9	841.1	1115.1	640.4	351.6	792.1	743.9	431.7	153.7	467.5	881.9	671.8	1188.1	979.3	771.5	395.4	0	313.1	806.3	1280.5
20	637.7	762.2	895	969.5	1428.2	953.5	664.7	1105.1	1057	744.7	466.8	702.7	695.5	358.8	875.1	666.3	458.4	202.9	313.1	0	493.2	1202
21	701.7	826.2	962.8	1033.5	1288.9	1148.9	985.7	1224.9	1377.9	1065.7	868.5	766.7	588.4	535.2	381.8	173	387.8	594.6	971.8	658.8	0	708.7
entrepôt	808	932.4	1296.3	939.1	580.2	1087.2	1376	1134.7	1429.6	1172	1186.7	872.9	454.4	757.5	658.6	656.6	792.4	1026	1340.4	1136.1	898.4	0

Tableau 5 : **matrice des distances.**

III.5. Résolution du problème :

Notre choix s'est porté aux méta-heuristique, plus précisément les algorithmes génétiques (AG) ainsi que le recuit simulé, en effet, les AG ont fait leurs preuves lors de nombreuses études notamment dans le domaine de la recherche opérationnelle ([23], [36], [38], [52], [53], [84], [69], [93]), et le recuit simulé est un algorithme facilement adaptable à n'importe quel problème en plus d'être léger et donc donne des résultats dans un temps court. La résolution va se dérouler en trois phases : la première étant la résolution d'un problème VRP classique avec les deux méta-heuristiques citées ; dans la deuxième phase nous allons assumer que la durée de livraison est linéairement croissante (détérioration), pour résoudre ce problème nous allons développer une méta-heuristique hybride à partir des deux méta-heuristiques précédentes ; pour finir nous allons chercher la meilleure position du RMA pour chaque véhicule. Une comparaison des méthodes utilisée sera faite sur la base de la valeur de la fonction objectif et du temps de traitement requis pour chaque algorithme, à chacune des phases de résolution.

Chapitre 3

5.1 Les méta-heuristiques :

le terme est divisé en deux mot dont l'origine est grecque : meta (voulant dire au-delà) et heuriskein (qui veut dire trouver). Les heuristiques sont des techniques de résolution itératives, spécialisées à un problème, et sans pour autant garantir une solution optimale. Quant aux méta-heuristiques, elles peuvent être adaptées à n'importe-quel problème, leur implémentation est naturellement plus difficile, elles sont donc utilisées dans les problèmes de la recherche opérationnelle, de l'ingénierie ou encore l'intelligence artificielle, dont les méthodes classiques du domaine ne donnent pas de solution acceptable.

5.2 Les algorithmes génétiques (AG) :

ce sont des méta-heuristiques fondés sur le mécanisme de la sélection naturelle et la génétique. Les algorithmes génétiques utilisent le même vocabulaire que celui de la théorie de l'évolution et de la génétique :

- Individu (solution potentielle)
- Population (ensemble de solutions)
- Génotype (représentation d'une solution)
- Gène (une partie du gène), parent, enfant, reproduction, croisement, mutation, …

5.3 Fonctionnement des AG :

5.3.1 Génération de la population initiale :

la première étape consiste à générer une population initiale qui se doit d'être suffisamment diversifiée et avec une taille considérable, et ce en guise de parcourir l'espace de l'état lors de la recherche se fasse dans un temps raisonnable.

5.3.2 Fonction d'adaptation :

durant la deuxième étape, nous allons mesurer la performance de chaque individu, on appel cette étape le calcul du fitness.

5.3.3 Sélection :

dans cette étape, nous allons choisir les parents qui nous donneront des enfants (les individus de la population de la prochaine génération). C'est un choix statistique des meilleurs individus de la population, cependant, l'opérateur de sélection doit être conçu pour donner également une chance aux mauvais éléments. On peut procéder à la sélection de trois façons distinctes : la sélection uniforme, la sélection binaire par tournois et la sélection par roulette. (Il est possible de choisir qu'une seule méthode)

5.3.3.1 Sélection binaire par tournois :

on choisit deux individus au hasard, on compare par la suite leurs fonctions d'adaptation pour en choisir le mieux adapté.

5.3.3.2 Sélection uniforme :

la sélection se fait de manière aléatoire et uniforme, telle que chaque individu est à une probabilité égale à $\frac{1}{Tpop}$.

5.3.3.3 Sélection par roulette : inspirée de la roue de la loterie où chaque individu est représenté par un secteur proportionnel à son fitness, dans cette méthode les individus les mieux évalués ont statistiquement plus de chance d'être sélectionnés, mais ça reste les individu mal adaptés d'être choisi.

5.3.4 Croisement : le croisement de deux parents génère deux enfant, cette étape favorise l'exploitation de l'espace de recherche et enrichit la diversité de la population. Il existe trois méthodes de croisement : croisement point, croisement deux points, et croisement uniforme.

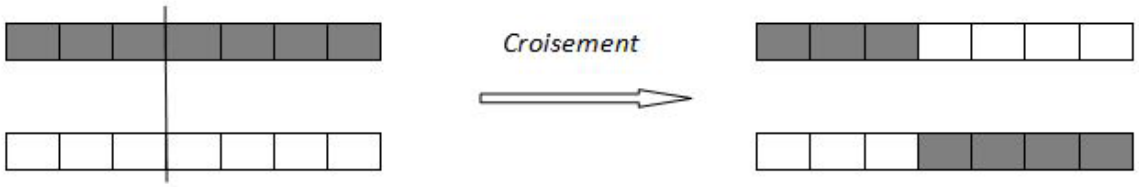


Figure 18 : **croisement un point.**

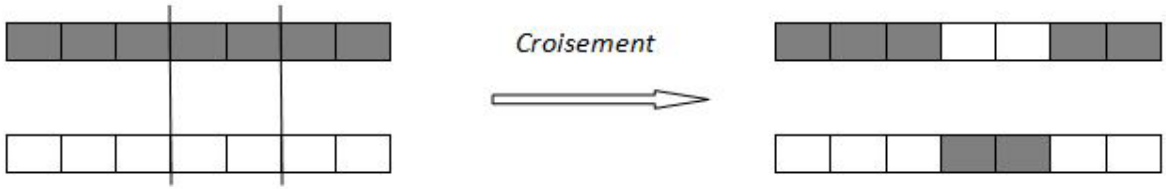


Figure 19 : **croisement deux points.**

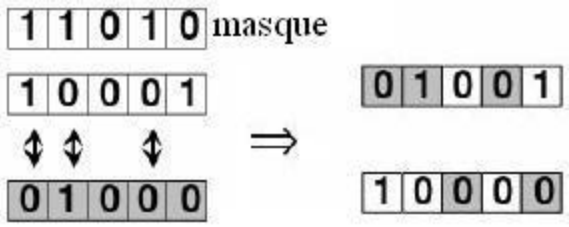


Figure 20 : **croisement uniforme.**

5.3.5 Mutation : dans cette étape, on procède à un changement mineur dans le code génétique pour un seul individu, et ce, pour une fin qui est d'introduire de la diversité (l'aspect aléatoire de la génétique) et ainsi, éviter de tomber dans des optimums locaux.

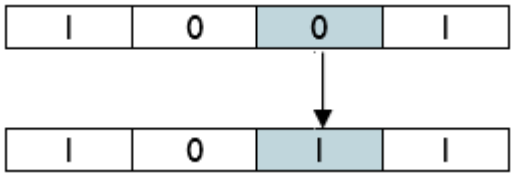


Figure 21 : **la mutation.**

5.3.6 L'élitisme : c'est la dernière étape, et elle consiste à enregistrer la meilleure solution en passant d'une population à une autre.

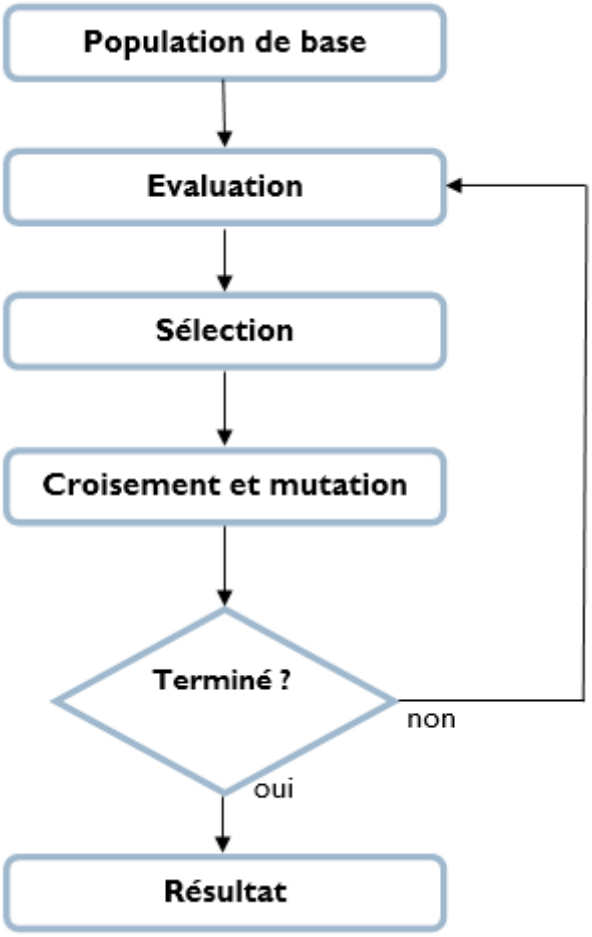


Figure 22 : **Le déroulement d'un algorithme génétique.**

5.4 Efficacité des AG : l'efficacité de l'algorithme est directement liée aux réglages des différents paramètres qui le caractérisent, voir :

- La taille de la population.
- Le nombre maximal des générations.
- La probabilité de mutation appelée pm.
- La probabilité de croisement nommée pc.

5.5 Recuit simulé : cette méta-heuristique a été mise au point par S. Kirkpatrick, C.D. Gelatt et M.P.Vecchi en 1983, elle s'inspire de la physique statistique et le refroidissement des métaux

5.6 Adaptation des algorithmes pour un problème VRP classique :

5.6.1 Codage de la solution : pour coder les solutions en guise d'appliquer nos algorithmes, nous avons choisi un système de codage liste, tel que nous représentons dans une vecteur, l'ordre que doit suivre chaque véhicule pour livrer ses clients ; nous séparons entre deux véhicules distincts par une case de valeur 0, les clients affectés à un véhicule donné seront donc successifs dans notre vecteur. Ci-dessous quelques exemples où l'on dispose de véhicule pour effectuer des livraisons pour 7 client, ces exemples vont nous servir pour mieux comprendre notre méthode de codage.

2	4	0	1	6	3	0	7	5

Tableau 6 : **codage d'une solution (1).**

0	1	2	5	0	3	4	7	6

Tableau 7 : **codage d'une solution (2).**

5	4	1	0	2	7	6	3	0

Tableau 8 : **codage d'une solution (3).**

Ayant donné des exemples où l'on considère une flotte de trois véhicules, nous avons choisi trois couleurs distinctes pour distinguer chaque véhicule, les clients affectés aux véhicule 1 sont colorés en bleu, les clients affectés au véhicule 2 sont coloré en vert et ceux du troisième véhicule en grena ; et comme expliqué en haut, nous séparons entre les clients de chaque véhicule avec une valeur 0. De ce, nous pouvons constater que dans le premier exemple le véhicule 1 devra livrer au client 2 puis au client 4, le véhicule deux va livrer au client 1 en premier lieu puis le client 6 et enfin le client 3, pour finir le véhicule 3 devra livrer au client 7 puis le client 5. Les deux autres exemples sont des cas particulier, tel que dans le deuxième exemple on constate que la première valeur du vecteur est 0, donc aucun client n'est affecté au véhicule un, le deuxième véhicule va livrer au client 1 puis le client 2 avant de finir avec le client 5, et le véhicule 3 va livrer au client 3, ensuite le client 4, puis 7 et en fin le client 6 ; concernant le troisième exemple, nous constatons que la dernière valeur du vecteur est un 0, ceci veut dire qu'aucun client ne sera affecté au véhicule 3, donc, le véhicule 1 aura à livrer le client 5 en premier, puis le client 4 et finir avec le client 1 ; le véhicule 2 va livrer pour le client 2, en outre le client 7, et 6 pour terminer avec le client 3.

5.6.2 Application de l'AG : ci-dessous le pseudocode de l'AG que nous avons déroulé pour la résolution de ce problème, et que nous avons codé sous python :

1. Définition de la fonction fitness.
2. Initialisation des paramètres de l'algorithme.
3. **Pop-P** : Initialisation de la population de base.
4. Evaluation de la population **Pop-P** : calcul du fitness à partir de la matrice qui nous rapporte le temps nécessaire pour se rendre d'un point à un autre.
5. **while** termination condition **do**.

// Naissance d'enfants.

6. Choix des parents avec la sélection binaire par tournois.
7. **Enfant-G** : croisement un point avec probabilité **Pr** dans la population **Pop-G**.
8. **Enfant-G** : mutation avec probabilité **Pm** des **Enfants-G**.
9. **Enfant-P** : décodage du génotype vers le phénotype de **Enfant-G**.

// Sélection naturelle.

10. **Pop-G** : sélection darwinienne des plus aptes {**Enfant-G**, **Pop-G**}.
11. end **while**.

Figure 23 : pseudocode de l'algorithme génétique.

Ci-dessous la solution générée par l'AG pour l'affectation des clients et l'ordonnancement des véhicules.

Tableau 9 : solution générée par notre AG pour un VRP classique.

Camion 2	16	14	18	19	20	21	12	10	6	11	7	2	1	17	5	3	8	9	4
Camion 3	**13**	**15**																	

Les informations que nous pouvons tirer de ces tableaux sont l'affectation des clients ainsi que les chemins que devront prendre les véhicules comme expliqué dans les exemples précédents, mais on peut aussi calculer le temps que va passer chaque véhicule pour effectuer toutes les livraisons qui lui ont été affiliées. Les résultats sont les suivant :

Calcul du fitness de la solution :

- Camion 1 : 0 U.T.
- Camion 2 : 9268.70 U.T.
- Camion 3 : 1303.40 U.T.

La fonction objectif est : 10572.10 U.T.

Le temps d'exécution de l'algorithme est : 2.04640s

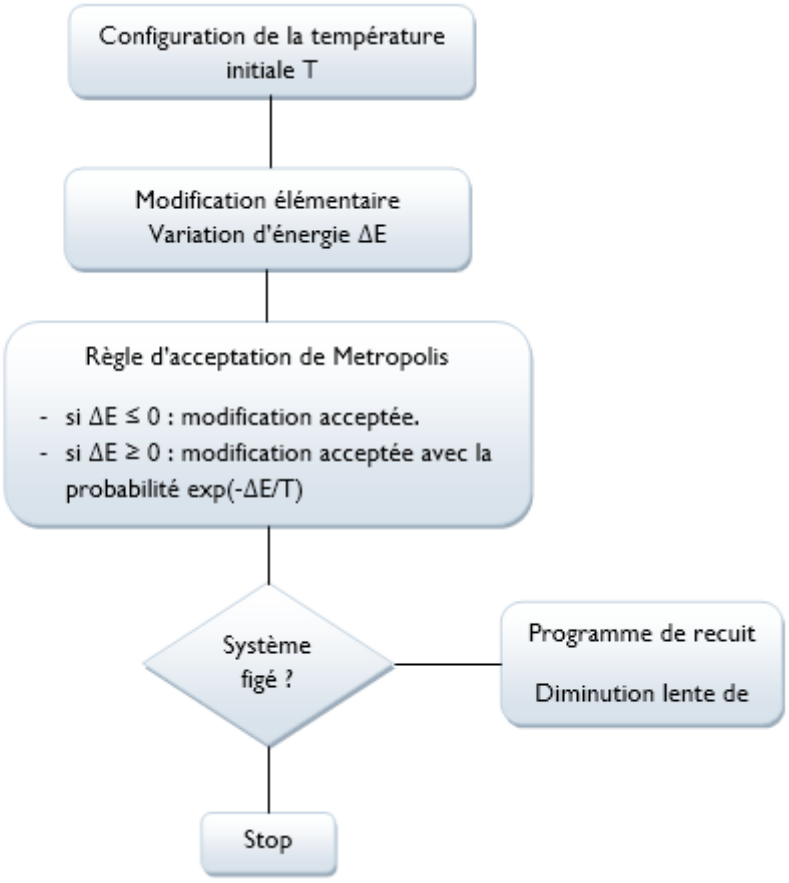


Figure 24 : **Le déroulement d'un algorithme de recuit simulé.**

5.6.3 Application du recuit simulé : ci-dessous le pseudocode du recuit simulé que nous avons déroulé pour la résolution de ce problème, et que nous avons codé sous python :

```
1.  i = i0                  #solution initiale
2.  T = T0                  #température initiale
3.  a = alpha               #coefficient de refroidissement.
4.  Tant que la condition d'arrêt n'est pas vérifiée :
5.      Tant que la fin du palier n'est pas atteinte :
6.          Générer une nouvelle solution j à partir de i
7.          ΔE = e(j) – e(i)
8.          si ΔE ≤ 0 alors :
9.              i = j
10.         sinon :
11.             i = j avec une probabilité de exp(- ΔE/T)
12.     T = T × a           #mise-à-jour de la température
```

Figure 15 : **pseudocode de l'algorithme recuit simulé.**

Ci-dessous la solution générée par l'algorithme de recuit simulé pour l'affectation des clients et l'ordonnancement des véhicules.

Tableau 10 : solution générée par notre algorithme de recuit simulé pour un VRP classique.

Camion2	5	9	8	6	4	3	20	16	18	17	21	15	12	2	11	7	19	10	14	1	13

Calcul du fitness de la solution :

- Camion 1 : 0 U.T.
- Camion 2 : 9682.10 U.T.
- Camion 3 : 0 U.T.

La fonction objectif est : 9682.10 U.T.

Le temps d'exécution de l'algorithme est : 0.0411s

Table 11 : comparaison entre les méthodes de résolution du VRP classique.

Les méta-heuristiques	*La fonction objectif*	*Le temps de traitement*
L'algorithme génétique	10572.10	2.04640
Le recuit simulé	9682.10	0.0411

Pour ce premier problème classique, il est clair que l'algorithme du recuit simulé est le mieux adapté, en effet, nous constatons que non seulement il retourne une meilleure solution que celle des AG, mais en plus de ça, le temps requis pour dérouler le programme est négligeable comparé à celui des AG.

5.7 Implémentation du modèle de détérioration dans notre VRP :

nous partons du principe que la détérioration est une fonction linéaire qui croit après chaque livraison effectuée, la détérioration est modélisée par la fonction det = **α***t, tel que **α** est une constante nommée coefficient de détérioration et qui va prendre comme valeur 1.2 dans notre exemple, le t représente une variable de croissance de la détérioration, elle sera incrémentée après chaque livraison effectuée. Ci-dessous le pseudo code du modèle de détérioration considéré dans notre étude.

```
1. α = 1.2              #coefficient de détérioration
2. t = 1                #initialisation de la température
3. Pour tous les point de livraison du véhicule (i) :
4.     Distance parcouru par V(i) += distance entre deux point de livraison x α*t
5.     t += 1
```

Figure 26 : pseudocode du modèle de détérioration.

A cette étape, nous constatons que les AG prennent un temps considérable pour retourner une solution plus au moins satisfaisante, tandis que le recuit simulé ne

fonctionne que pour un nombre d'itération calculé en milliers, et donc il ne converge pas forcément vers la solution optimale, la pertinence de la solution que donne cet algorithme est donc remise en question ; c'est dans cette optique que nous nous sommes investi pour développer un autre algorithme hybride, à partir des deux premiers algorithmes, tel que la meilleure solution que trouveras l'algorithme génétique sera pris comme solution initiale pour la méta-heuristique du recuit simulé. Dans la figure qui suit, vous trouverez le pseudocode de la méta-heuristique hybride que nous avons développé pour résoudre le problème de détérioration.

```
Définition de la fonction fitness                    #modèle de détérioration.
Roulement de l'algorithme génétique pour N générations :
    Sélection binaire par tournois
    Croisement
    Mutation
    Elitisme
sol_initiale = meilleure solution donnée par l'AG
r = sol_initiale
s = sol_initiale
sol = sol_initiale
Faire pour un nombre d'itération M :
    Permuter deux valeurs du vecteur r
    si fitness r < fitness s :
        s = r
    si non :
        s = r avec une probabilité exp((fitness s − fitness r)/T)
    si fitness r < fitness sol :
        sol = r
    Mettre-à-jour T
Retourner sol
```

Figure 27 : pseudocode de la méta-heuristique hybride développée.

Pour démontrer l'efficacité de notre algorithme, nous allons faire une comparaison entre les trois méta-heuristiques, les résultats sont reportés dans le tableau suivant.

Table 12 : solution générée par l'AG pour le modèle de détérioration.

Camion 1	**9**	**15**	**2**	**1**	**12**	**17**		
Camion 2	**16**	**3**	**11**	**18**	**21**	**14**	**13**	
Camion 3	**20**	**7**	**19**	**10**	**6**	**8**	**4**	**5**

Table 13 : solution générée par le recuit simulé pour le modèle de détérioration.

Camion 1	10	18	5	17	13	14	16	21	15
Camion 2	20	2	12	11	19				
Camion 3	9	7	3	1	4	6	8		

Table 14 : solution générée par l'algorithme hybride pour le modèle de détérioration.

Camion 1	14	1	15	13	17	16	21
Camion 2	8	2	20	18	12	11	19
Camion 3	5	9	4	6	7	10	3

Table 15 : comparaison entre les méthodes de résolution du modèle de détérioration.

Les méta-heuristiques	*La fonction objectif*	*Le temps de traitement*
L'algorithme génétique	35816.8	81.0464
Le recuit simulé	34784.58	34.9198
La méta-heuristique hybride	29649.36	81.0905

Dans ce cas, nous constatons que la méta-heuristique que nous avons développé donne une meilleure solution que les deux autres algorithmes, chose justifiée car la meilleure solution que génère l'AG sera par la suite la solution de base que va optimiser l'algorithme génétique, par contre, cette optimisation sera sanctionnée par un temps de traitement plus considérable vu que les AG tardent lors du déroulement du programme.

5.8 Rate modifying activity (RMA) :

dans cette partie, nous proposons d'appliquer une activité modificatrice de taux de rendement, tout en essayant de trouver la meilleure position du RMA, pour ce faire, nous avons appliqué quelques modifications sur la fonction de notre programme permettant le calcul de la fonction objectif, de tel sorte que la détérioration est réinitialisée à 0 après chaque RMA appliquée, chaque véhicule a le droit de faire une pause en guise de repos (RMA), qu'il choisit de faire ou pas selon le cas, pour résoudre le problème de la position du RMA, et de si une RMA nous permet une optimisation ou pas, nous avons développé un modèle permettant de tester tous les cas possibles pour chaque véhicule, et qui nous retourne à la fin la meilleure solution avec les meilleures position du RMA pour minimiser notre fonction objectif. Ci-dessous un exemple illustré permettant de comprendre le fonctionnement de notre programme :

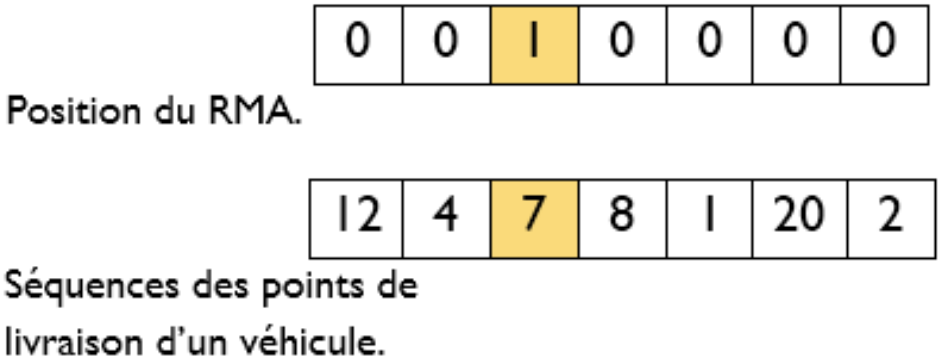


Figure 28 : **position d'un RMA.**

Dans ce cas, la position du RMA est la 3ème, elle serait appliquée dans le point de livraison N°7, ceci dit, la détérioration entre 12 et 4 serait égale à 1.2, puis entre 4 et 7 2.2 pour qu'elle soit réinitialisée à sa valeur initiale après la livraison dans le point de livraison 7, une valeur de 200 sera ajoutée à la fonction objectif afin de comptabiliser l'arrêt (pause) effectué pour faire notre RMA, et après le 8ème point la détérioration va continuer à croitre après chaque livraison.

Ci-dessous le pseudocode de la fonction permettant le calcul du fitness avec considération du RMA :

```
a = 1.2                                  #initialisation du coefficient de détérioration
t = 1                                    #initialisation de la variable de détérioration
RMA = 0
Pour tous les points de livraison du véhicule (i) faire :
    RMA += 1
    Pour tous les points de livraison du véhicule (i) faire :
        si l'indice du point = RMA
            t = 1
        Distance parcourue par V(i) = distance entre deux point * a * t
        t += 1
```

Figure 29 : **pseudocode du programme calculant la meilleure position du RMA.**

Nous utilisons nos trois méta-heuristiques afin de comparer entre elle et de savoir laquelle d'entre elles est la mieux adaptée à ce problème, les résultats sont rapportés dans le tableau suivant :

Tableau 16 : comparaison des solutions avec RMA pour les trois algorithmes.

Les méta-heuristiques	*La fonction objectif*	*RMA1*	*RMA2*	*RMA3*
L'algorithme génétique	23420.83	1	10	18
Le recuit simulé	25014.42	5	12	1
La méta-heuristique hybride	20185.32	13	18	6

Nous remarquons que pour appliquer une RMA, l'algorithme du recuit simulé est le moins adapté, bien qu'il soit bien adapté pour les deux problèmes considérés précédemment, cependant, nous constatons que la RMA réduit considérablement la valeur de la fonction objectif pour les trois algorithmes, donc nous avons bel et bien réussi à optimiser notre problème de VRP à effet de détérioration avec les RMA.

III.6. Conclusion :

Dans ce chapitre, nous avons introduit le problème VRP avec considération de l'effet de détérioration, nous justifions l'effet de détérioration dans la tournée de véhicules par la fatigue du chauffeur régie par le cumule d'heures de travail. Nous avons d'abord faite une formulation du problème, puis nous avons présenté les algorithmes génétiques et la méta-heuristique du recuit simulé qui nous ont permises de résoudre un problème VRP classique pour un premier temps, en suite, nous avons introduit le problème de VRP avec détérioration ayant considéré une détérioration croissante linéairement avec un facteur de 1.2 après chaque livraison effectuée, et par la même occasion nous avons présenté une méta-heuristique développée à partir de l'hybridation des deux premières, et en fin, nous avons cherché à trouver la meilleur position du RMA pour chaque véhicule pour une fin qui est de minimiser notre fonction objectif. Une comparaison sur la base de la valeur du fitness des solutions générées par chaque algorithme, et du temps de traitement de chacun des trois algorithmes développés a été faite en guise de comparaison, et de trouver la méta-heuristique la mieux adaptée à ce problème. Nous constatons que le recuit simulé étant le plus rapide en ce qui concerne le temps requis pour converger, mais n'est pas aussi efficace que les deux autres algorithmes lorsque nous considérons une RMA ; les AG bien qu'ils soit bien adaptés pour le problème avec RMA, mais prennent un temps considérable lors du déroulement du programme ; par contre, l'hybridation de ces deux méta-heuristiques permet de trouver des solutions meilleures pour la résolutions des problème VRP avec détérioration, et avec position de RMA.

Conclusion générale.

conclusion générale

Dans ce projet, nous avons introduit le problème VRP avec effet de détérioration, nous avons justifié l'effet de détérioration dans la tournée de véhicules avec la fatigue du chauffeur régie par le cumule d'heures de travail. Après s'être intéressés aux problèmes d'ordonnancement vu leur importance pour les industriels, l'effet de détérioration représentaient un sujet intéressant pour nous, vu que cette contrainte n'est pas souvent prise en considération dans les études de cas, malgré qu'elle est très fréquente dans la réalité ; mais après avoir de nombreuse recherches à ce sujet, nous constatons que l'effet de détérioration dans toutes ses formes a été considéré dans la plus part du temps dans la production, c'est de là que nous nous somme motivés à introduire la contrainte de détérioration aux problèmes de transport, notamment le problème VRP vu l'importance d'optimiser la tournée de véhicules pour les industriels.

De ce, en utilisant trois méta-heuristiques pour résoudre notre problème, et qui sont : (**i**) l'algorithme génétique, (**ii**) le recuit simulé et (**iii**) une hybridation des deux algorithmes précédents, nous avons commencé par leur adaptation pour une fin qui est la résolution d'un problème VRP classique, puis nous les avons implémentés dans le modèle de détérioration élaboré, et enfin dans le modèle de détérioration avec RMA. Et pour finir, nous avons faite une comparaison entre nos trois algorithmes afin de savoir lequel est mieux adapté pour ce genre de problème.

En guise d'aspiration, les futurs projets qui porteront sur l'optimisation de la tournée de véhicules considérant l'effet de détérioration peuvent considérer d'autre contraintes tel que la capacité des véhicules et les fenêtre de temps, il serait aussi intéressant de comparer les solutions données par les méta-heuristiques avec d'autres méthodes en l'occurrence : le machine learning (en utilisant par exemple l'algorithme de clustering **K-means**), la simulation avec les multi-agents, ou encore avec un solver tel CPlex. Tout comme on pourrait considérer des moyens de manutention tels les AGV dans une usine, où l'on considérera le temps de traitement des machines.

Toutefois, nous espérons que ce modeste travail puisse apporter un plus et constituer un support supplémentaire aux promotions à venir.

Liste des références.

Références

1- www.journaldunet.fr
2- Wikipédia
3- www.sage.com
4- cloudfront.net
5- Génie-Alimentaire.com
6- Livre Factory physics. Wallace J. Hopp, Mark L. Spearman.
7- Support du cours d'ordonnancement. E.S.S.A.T. Mehdi SOUIER.
8- R. L. Graham, E. L. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, Annals of Discrete Mathematics 5 (1979), 287-326.
9- Support de cours département des mathématiques et génie industriel à polytechniques Montreal-Canada.
10- Silverwschatz A., Galvin P., Gagné G., Principles appliqués des systèmes d'exploitation avec Java, Vuibert, Paris, France, 2002.
11- Principes appliqués des systèmes d'exploitation. Avec Java. G. Gagne, Peter Galvin, Abraham Silberschatz. mars 2008.
12- Cours de techniques d'ordonnancement. G.BAVIER voir sos-math.fr
13- Livre de Facilities Design. Sunderesh S. Heragu. https://doi.org/10.1201/9781315382647
14- Tigane M., Dahane M., Boudhar M., Multiobjective approach for deteriorating jobs scheduling for a sustainable manufacturing system (2018). https://doi.org/10.1007/s00170-018-3043-1
15- P. Lopez et F. Roubellat, Ordonnancement de la production, Hermes Science Europe Ltd, 2001.
16- B. Montreuil, Fractal layout organization for job shop, int. j. prod. res., 1999, vol. 37, no. 3, 501-521.
17- Molla, R. S. (2018, May). A study on Manufacturing of Deformed Bar (G 60-400W) at Elite Iron and Steel Industries. Retrieved from Research Gate: https://www.researchgate.net/publication/325178996_A_study_on_Manufacturing_of_Deformed_Bar_G_60-400W_at_Elite_Iron_and_Steel_Industries
18- Slack, Chambers, S and Johnston, R (2009), livre Operations Management.
19- https://elearn.univ-tlemcen.dz/pluginfile.php/76346/mod_resource/content/1/chapitre2.pdf
20- Tigane, Meriem. Ordonnancement et gestion de l'énergie. Diss. 2021.
21- WOODWARD, JOAN. Industrial Organization: Theory and Practice, 1965; Tom BURNS and GM STALKER. The Management of Innovation, 1961.
22- Livre Méthodes mathématiques d'organisation et de planification de la production. Leonid Kantorovič 1939
23- LIU, Cheng-Hsiang et HUANG, Ding-Hsiang. Reduction of power consumption and carbon footprints by applying multi-objective optimisation via genetic algorithms. International Journal of Production Research, 2014, vol. 52, no 2, p. 337-352.
24- RABADI, Ghaith, MORAGA, Reinaldo J., et AL-SALEM, Ameer. Heuristics for the unrelated parallel machine scheduling problem with setup times. Journal of Intelligent Manufacturing, 2006, vol. 17, no 1, p. 85-97.

25- ARNAOUT, Jean-Paul, RABADI, Ghaith, et MUSA, Rami. A two-stage ant colony optimization algorithm to minimize the makespan on unrelated parallel machines with sequence-dependent setup times. Journal of Intelligent Manufacturing, 2010, vol. 21, no 6, p. 693-701.
26- ARNAOUT, Jean-Paul, MUSA, Rami, et RABADI, Ghaith. A two-stage Ant Colony optimization algorithm to minimize the makespan on unrelated parallel machines—part II: enhancements and experimentations. Journal of Intelligent Manufacturing, 2014, vol. 25, no 1, p. 43-53.
27- CHARALAMBOUS, Christoforos et FLESZAR, Krzysztof. Variable neighborhood descent for the unrelated parallel machine scheduling problem. International Journal on Artificial Intelligence Tools, 2012, vol. 21, no 04, p. 1240019.
28- FANJUL-PEYRO, Luis et RUIZ, Rubén. Scheduling unrelated parallel machines with optional machines and jobs selection. Computers & Operations Research, 2012, vol. 39, no 7, p. 1745-1753.
29- GAREY, Michael R. et JOHNSON, David S. A Guide to the Theory of NP-Completeness. Computers and intractability, 1990, p. 37-79.
30- HARIRI, A. M. A. et POTTS, Chris N. Heuristics for scheduling unrelated parallel machines. Computers & operations research, 1991, vol. 18, no 3, p. 323-331.
31- KUMAR, K. S., SELLADURAI, V., RAJA, K., et al. Ant colony approach for makespan minimization on unrelated parallel machines. International Journal of Engineering Science and Technology, 2011, vol. 3.
32- SHCHEPIN, Evgeny V. et VAKHANIA, Nodari. An optimal rounding gives a better approximation for scheduling unrelated machines. Operations Research Letters, 2005, vol. 33, no 2, p. 127-133.
33- ZHANG, Xianzhao, XU, Dachuan, DU, Donglei, et al. Approximate algorithms for unrelated machine scheduling to minimize makespan. Journal of Industrial & Management Optimization, 2016, vol. 12, no 2, p. 771.
34- CHEN, Jeng-Fung. Unrelated parallel machine scheduling with secondary resource constraints. The International Journal of Advanced Manufacturing Technology, 2005, vol. 26, no 3, p. 285-292.
35- YANG-KUEI, Lin et CHI-WEI, Lin. Dispatching rules for unrelated parallel machine scheduling with release dates. The International Journal of Advanced Manufacturing Technology, 2013, vol. 67, no 1, p. 269-279.
36- CHANG, Pei-Chann et CHEN, Shih-Hsin. Integrating dominance properties with genetic algorithms for parallel machine scheduling problems with setup times. Applied Soft Computing, 2011, vol. 11, no 1, p. 1263-1274.
37- DE PAULA, Mateus Rocha, RAVETTI, Martín Gómez, MATEUS, Geraldo Robson, et al. Solving parallel machines scheduling problems with sequence-dependent setup times using variable neighbourhood search. IMA Journal of Management Mathematics, 2007, vol. 18, no 2, p. 101-115.
38- YILMAZ EROGLU, Duygu, OZMUTLU, H. Cenk, et OZMUTLU, Seda. Genetic algorithm with local search for the unrelated parallel machine scheduling problem with

sequence-dependent set-up times. International Journal of Production Research, 2014, vol. 52, no 19, p. 5841-5856.
39- EZUGWU, Absalom E., ADELEKE, Olawale J., et VIRIRI, Serestina. Symbiotic organisms search algorithm for the unrelated parallel machines scheduling with sequence-dependent setup times. PloS one, 2018, vol. 13, no 7, p. e0200030.
40- NIKABADI, M. et NADERI, Reihaneh. A hybrid algorithm for unrelated parallel machines scheduling. International Journal of Industrial Engineering Computations, 2016, vol. 7, no 4, p. 681-702.
41- AFZALIRAD, Mojtaba et REZAEIAN, Javad. Resource-constrained unrelated parallel machine scheduling problem with sequence dependent setup times, precedence constraints and machine eligibility restrictions. Computers & Industrial Engineering, 2016, vol. 98, p. 40-52.
42- DAMODARAN, Purushothaman, DIYADAWAGAMAGE, Don Asanka, GHRAYEB, Omar, et al. A particle swarm optimization algorithm for minimizing makespan of nonidentical parallel batch processing machines. The International Journal of Advanced Manufacturing Technology, 2012, vol. 58, no 9, p. 1131-1140.
43- EBRAHIMI, E. et REZAEIAN, J. Unrelated parallel machines scheduling with the effect of aging and learning under multi maintenance activities. Manufacturing Science and Technology, 2015, vol. 3, no 1, p. 25-31.
44- ARTHANARI, T. S. et RAMAMURTHI, K. G. A branch and bound algorithm for sequencing n jobs on m parallel processors. Opsearch, 1970, vol. 7, p. 147-156.
45- BANSAL, Nikhil, BUNDE, David P., CHAN, Ho-Leung, et al. Average rate speed scaling. Algorithmica, 2011, vol. 60, no 4, p. 877-889.
46- VIGNIER, A., BILLAUT, J. C., et PROUST, C. Scheduling problems of hybrid flowshop type: state of the art. RAIRO. Operations Research, 1999, vol. 33, no 2, p. 117-183.
47- BASKAR, A. et XAVIOR, M. Anthony. A new Heuristic algorithm using Pascal's triangle to determine more than one sequence having optimal/near optimal make span in flow shop scheduling problems. International Journal of Computer Applications, 2012, vol. 975, p. 8887.
48- CHAKRABORTY, Uday Kumar et LAHA, Dipak. An improved heuristic for permutation flowshop scheduling. International Journal of Information and communication technology, 2007, vol. 1, no 1, p. 89-97.
49- MODRÁK, Vladimír et PANDIAN, R. Sudhakara. Flow shop scheduling algorithm to minimize completion time for n-jobs m-machines problem. Tehnički vjesnik, 2010, vol. 17, no 3, p. 273-278.
50- NOWICKI, Eugeniusz et SMUTNICKI, Czesław. A fast tabu search algorithm for the permutation flow-shop problem. European Journal of Operational Research, 1996, vol. 91, no 1, p. 160-175.
51- STÜTZLE, Thomas, et al. An ant approach to the flow shop problem. In : Proceedings of the 6th European Congress on Intelligent Techniques & Soft Computing (EUFIT'98). Aachen : Verlag Mainz, Wissenschaftsverlag, 1998. p. 1560-1564.

Références

52- TSENG, Lin-Yu et LIN, Ya-Tai. A hybrid genetic local search algorithm for the permutation flowshop scheduling problem. European Journal of Operational Research, 2009, vol. 198, no 1, p. 84-92.

53- CHAUDHRY, Imran A., AHMED, Riaz, et KHAN, Abdul Munem. Genetic Algorithm to minimize flowtime in a no-wait flowshop scheduling problem. In : IOP Conference Series: Materials Science and Engineering. IOP Publishing, 2014. p. 012007.

54- NAILWAL, K., GUPTA, D., et JEET, K. Heuristics for no-wait flow shop scheduling problem. International Journal of Industrial Engineering Computations, 2016, vol. 7, no 4, p. 671-680.

55- YE, Honghan, LI, Wei, ABEDINI, Amin, et al. An effective and efficient heuristic for no-wait flow shop production to minimize total completion time. Computers & Industrial Engineering, 2017, vol. 108, p. 57-69.

56- SAPKAL, Sagar U. et LAHA, Dipak. A heuristic for no-wait flow shop scheduling. The International Journal of Advanced Manufacturing Technology, 2013, vol. 68, no 5, p. 1327-1338.

57- WANG, Ling, PAN, Quan-Ke, et TASGETIREN, M. Fatih. Minimizing the total flow time in a flow shop with blocking by using hybrid harmony search algorithms. Expert Systems with Applications, 2010, vol. 37, no 12, p. 7929-7936.

58- BALAJI, A. N. et PORSELVI, S. Artificial immune system algorithm and simulated annealing algorithm for scheduling batches of parts based on job availability model in a multi-cell flexible manufacturing system. Procedia Engineering, 2014, vol. 97, p. 1524-1533.

59- WANG, Chuyang, LI, Xiaoping, et WANG, Qian. Accelerated tabu search for no-wait flowshop scheduling problem with maximum lateness criterion. European Journal of Operational Research, 2010, vol. 206, no 1, p. 64-72.

60- TSENG, Chao-Tang, LIAO, Ching-Jong, et LIAO, Tai-Xiang. A note on two-stage hybrid flowshop scheduling with missing operations. Computers & Industrial Engineering, 2008, vol. 54, no 3, p. 695-704.

61- GUIRCHOUN, Samuel, MARTINEAU, Patrick, et BILLAUT, J.-C. Total completion time minimization in a computer system with a server and two parallel processors. Computers & Operations Research, 2005, vol. 32, no 3, p. 599-611.

62- RIANE, Fouad, ARTIBA, Abdelhakim, et ELMAGHRABY, Salah E. Sequencing a hybrid two-stage flowshop with dedicated machines. International Journal of Production Research, 2002, vol. 40, no 17, p. 4353-4380.

63- RIANE, Fouad, ARTIBA, Abdelhakim, et ELMAGHRABY, Salah E. A hybrid three-stage flowshop problem: efficient heuristics to minimize makespan. European Journal of Operational Research, 1998, vol. 109, no 2, p. 321-329.

64- HEKMATFAR, Masood, GHOMI, SMT Fatemi, et KARIMI, Behrooz. Two stage reentrant hybrid flow shop with setup times and the criterion of minimizing makespan. Applied Soft Computing, 2011, vol. 11, no 8, p. 4530-4539.

65- DONG, Jianming, TONG, Weitian, LUO, Taibo, et al. An FPTAS for the parallel two-stage flowshop problem. Theoretical computer science, 2017, vol. 657, p. 64-72.

66- AURICH, Paul, NAHHAS, Abdulrahman, REGGELIN, Tobias, et al. Simulation-based optimization for solving a hybrid flow shop scheduling problem. In : 2016 Winter Simulation Conference (WSC). IEEE, 2016. p. 2809-2819.
67- LEI, Deming et GUO, Xiuping. Hybrid flow shop scheduling with not-all-machines options via local search with controlled deterioration. Computers & Operations Research, 2016, vol. 65, p. 76-82.
68- YI, Wenchao, GAO, Liang, ZHOU, Yinzhi, et al. Differential evolution algorithm with variable neighborhood search for hybrid flow shop scheduling problem. In : 2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD). IEEE, 2016. p. 233-238.
69- SHIAU, Der-Fang, CHENG, Shu-Chen, et HUANG, Yueh-Min. Proportionate flexible flow shop scheduling via a hybrid constructive genetic algorithm. Expert systems with applications, 2008, vol. 34, no 2, p. 1133-1143.
70- PACHECO, Gina M. Galindo, POLO, Luis E. Ramirez, et NIÑO, Vanessa P. Manotas. Flexible Flowshop problem minimizing total flow time and makespan using Evolutionary Algorithm. In : Proc. of the Latin American and Caribbean Conference for Engineering and Technology (LACCEI). 2013. p. 1-9.
71- PAN, Quan-Ke et DONG, Yan. An improved migrating birds optimisation for a hybrid flowshop scheduling with total flowtime minimisation. Information Sciences, 2014, vol. 277, p. 643-655.
72- MOUSAVI, S. M., MOUSAKHANI, M., et ZANDIEH, M. Bi-objective hybrid flow shop scheduling: a new local search. The International Journal of Advanced Manufacturing Technology, 2013, vol. 64, no 5, p. 933-950.
73- GUPTA, Jatinder ND et GUPTA, Sushil K. Single facility scheduling with nonlinear processing times. Computers & Industrial Engineering, 1988, vol. 14, no 4, p. 387-393.
74- BROWNE, Sid et YECHIALI, Uri. Scheduling deteriorating jobs on a single processor. Operations Research, 1990, vol. 38, no 3, p. 495-498.
75- WANG, Ji-Bo, NG, Chi To, et CHENG, TC Edwin. Single-machine scheduling with deteriorating jobs under a series–parallel graph constraint. Computers & Operations Research, 2008, vol. 35, no 8, p. 2684-2693.
76- KUO, Wen-Hung et YANG, Dar-Li. Parallel-machine scheduling with time dependent processing times. Theoretical Computer Science, 2008, vol. 393, no 1-3, p. 204-210.
77- LI, Shisheng et YUAN, Jinjiang. Parallel-machine scheduling with deteriorating jobs and rejection. Theoretical Computer Science, 2010, vol. 411, no 40-42, p. 3642-3650.
78- LIN, Shih-Wei et YING, Kuo-Ching. A multi-point simulated annealing heuristic for solving multiple objective unrelated parallel machine scheduling problems. International Journal of Production Research, 2015, vol. 53, no 4, p. 1065-1076.
79- LI, Wenhan, CHEN, Xiaolong, LI, Junqing, et al. An improved iterated greedy algorithm for distributed robotic flowshop scheduling with order constraints. Computers & Industrial Engineering, 2022, vol. 164, p. 107907.
80- LI, Jun-Qing, SONG, Mei-Xian, WANG, Ling, et al. Hybrid artificial bee colony algorithm for a parallel batching distributed flow-shop problem with deteriorating jobs. IEEE transactions on cybernetics, 2019, vol. 50, no 6, p. 2425-2439.

Références

81- RUIZ, Rubén, PAN, Quan-Ke, et NADERI, Bahman. Iterated Greedy methods for the distributed permutation flowshop scheduling problem. Omega, 2019, vol. 83, p. 213-222.

82- LEI, Deming, YUAN, Yue, CAI, Jingcao, et al. An imperialist competitive algorithm with memory for distributed unrelated parallel machines scheduling. International Journal of Production Research, 2020, vol. 58, no 2, p. 597-614.

83- LEE, C.-Y. et LEON, V. Jorge. Machine scheduling with a rate-modifying activity. European Journal of Operational Research, 2001, vol. 128, no 1, p. 119-128.

84- WOO, Young-Bin, JUNG, Sunwoong, et KIM, Byung Soo. A rule-based genetic algorithm with an improvement heuristic for unrelated parallel machine scheduling problem with time-dependent deterioration and multiple rate-modifying activities. Computers & Industrial Engineering, 2017, vol. 109, p. 179-190.

85- BACHMAN, Aleksander, JANIAK, Adam, et KOVALYOV, Mikhail Y. Minimizing the total weighted completion time of deteriorating jobs. Information Processing Letters, 2002, vol. 81, no 2, p. 81-84.

86- JI, Min et CHENG, TC Edwin. Parallel-machine scheduling with simple linear deterioration to minimize total completion time. European Journal of Operational Research, 2008, vol. 188, no 2, p. 342-347.

87- WANG, Ji-Bo, SUN, Lin-Hui, et SUN, Lin-Yan. Single-machine total completion time scheduling with a time-dependent deterioration. Applied Mathematical Modelling, 2011, vol. 35, no 3, p. 1506-1511.

88- WANG, Xiao-Yuan et WANG, Jian-Jun. Scheduling deteriorating jobs with a learning effect on unrelated parallel machines. Applied Mathematical Modelling, 2014, vol. 38, no 21-22, p. 5231-5238.

89- ZHU, Hui, LI, Min, et ZHOU, Zhangjin. Machine scheduling with deteriorating and resource-dependent maintenance activity. Computers & Industrial Engineering, 2015, vol. 88, p. 479-486.

90- YANG, Dar-Li et YANG, Suh-Jenq. Unrelated parallel-machine scheduling problems with multiple rate-modifying activities. Information Sciences, 2013, vol. 235, p. 280-286.

91- WANG, Xiao-Yuan et WANG, Ming-Zheng. Single machine common flow allowance scheduling with a rate-modifying activity. Computers & Industrial Engineering, 2010, vol. 59, no 4, p. 898-902.

92- ZHAO, Chuan-Li, TANG, Heng-Yong, et CHENG, Cong-Dian. Two-parallel machines scheduling with rate-modifying activities to minimize total completion time. European Journal of Operational Research, 2009, vol. 198, no 1, p. 354-357.

93- DO CHUNG, Byung et KIM, Byung Soo. A hybrid genetic algorithm with two-stage dispatching heuristic for a machine scheduling problem with step-deteriorating jobs and rate-modifying activities. Computers & Industrial Engineering, 2016, vol. 98, p. 113-124.

94- JOO, Cheol Min et KIM, Byung Soo. Machine scheduling of time-dependent deteriorating jobs with determining the optimal number of rate modifying activities and the position of the activities. Journal of Advanced Mechanical Design, Systems, and Manufacturing, 2015, vol. 9, no 1, p. JAMDSM0007-JAMDSM0007.

Références

95- WANG, Jian-Jun, WANG, Ji-Bo, et LIU, Feng. Parallel machines scheduling with a deteriorating maintenance activity. Journal of the Operational Research Society, 2011, vol. 62, no 10, p. 1898-1902.

96- WOO, Young-Bin et KIM, Byung Soo. A Study on Identical Parallel Machine Scheduling with Deterioration and Rate-Modifying Activities. ICIC express letters. Part B, Applications: an international journal of research and surveys, 2016, vol. 7, no 10, p. 2117-2122.

97- YANG, Dar-Li et YANG, Suh-Jenq. Unrelated parallel-machine scheduling problems with multiple rate-modifying activities. Information Sciences, 2013, vol. 235, p. 280-286.

98- YANG, Suh-Jenq. Unrelated parallel-machine scheduling with deterioration effects and deteriorating multi-maintenance activities for minimizing the total completion time. Applied Mathematical Modelling, 2013, vol. 37, no 5, p. 2995-3005.

99- ZHANG, Xingong, YIN, Yunqiang, et WU, Chin-Chia. Scheduling with non-decreasing deterioration jobs and variable maintenance activities on a single machine. Engineering Optimization, 2017, vol. 49, no 1, p. 84-97.

100- ZHAO, Chuan-Li, TANG, Heng-Yong, et CHENG, Cong-Dian. Two-parallel machines scheduling with rate-modifying activities to minimize total completion time. European Journal of Operational Research, 2009, vol. 198, no 1, p. 354-357.

101- ZHU, Hui, LI, Min, et ZHOU, Zhangjin. Machine scheduling with deteriorating and resource-dependent maintenance activity. Computers & Industrial Engineering, 2015, vol. 88, p. 479-486.

102- LODREE JR, Emmett J. et GEIGER, Christopher D. A note on the optimal sequence position for a rate-modifying activity under simple linear deterioration. European Journal of Operational Research, 2010, vol. 201, no 2, p. 644-648.

103- FISZMAN, Shir et MOSHEIOV, Gur. Minimizing total load on a proportionate flowshop with position-dependent processing times and job-rejection. Information Processing Letters, 2018, vol. 132, p. 39-43.

104- AGNETIS, Alessandro et MOSHEIOV, Gur. Scheduling with job-rejection and position-dependent processing times on proportionate flowshops. Optimization Letters, 2017, vol. 11, no 4, p. 885-892.

105- WANG, Ji-Bo. Flow shop scheduling jobs with position-dependent processing times. Journal of Applied Mathematics and Computing, 2005, vol. 18, no 1, p. 383-391.

106- BADIRU, Adedeji B. Computational survey of univariate and multivariate learning curve models. IEEE transactions on Engineering Management, 1992, vol. 39, no 2, p. 176-188.

107- SEKKAL, Norelhouda et BELKAID, Fayçal. A multi-objective simulated annealing to solve an identical parallel machine scheduling problem with deterioration effect and resources consumption constraints. Journal of Combinatorial Optimization, 2020, vol. 40, no 3, p. 660-696.

Résumé :

Dans ce projet, nous nous sommes intéressés au développement de trois méta-heuristiques à savoir les algorithmes génétiques, le recuit simulé et un algorithme hybride à partir des deux premiers, et ce afin d'optimiser la tournée de véhicules de livraison tout en considérant l'effet de détérioration qui retarde la date de livraison en ralentissant le véhicule chargé du transport. Nous résolvons d'abord un problème VRP classique, en suite nous développons un modèle de détérioration linéaire que nous inculquons au problème VRP, et pour finir, nous cherchons la position idéale pour une activité modificatrice de taux de rendement. Nous comparons notamment les solutions générées par nos trois algorithmes afin de déterminer quelle est la méta-heuristique la plus adaptée à ce problème.

Mots clefs : recherche opérationnelle, ordonnancement, logistique urbaine, transport, VRP, méta-heuristiques, algorithmes génétiques, recuit simulé, méta-heuristique hybride.

Abstract:

In this project, we are interested in developing three meta-heuristics, namely genetic algorithms, simulated annealing and a hybrid algorithm from the first two, in order to optimize the delivery vehicle routing while considering the deterioration effect that delays the delivery date by slowing down the vehicle in charge of the transport. We first solve a classical VRP, then we develop a linear deterioration model that we include in the VRP problem, and finally, we look for the ideal position for a rate modifying activity. Then we compare the solutions generated by our three algorithms in order to determine which meta-heuristic is best suited to this problem.

Keywords: operations research, scheduling, urban logistics, transportation, VRP, meta-heuristics, genetic algorithms, simulated annealing, hybrid meta-heuristics.

ملخص:

في هذا المشروع، ركزنا على تطوير ثلاث meta-heuristics: الخوارزميات الجينية، و simulated annealing وخوارزمية هجينة من الأولين، وهذا من أجل تحسين جولة مركبة التوصيل مع مراعاة تأثير التدهور الذي يؤخر تاريخ التسليم بسبب إبطاء السيارة المحملة بالنقل. نقوم أولاً بحل مشكلة VRP كلاسيكية، ثم نطور نموذج تدهور خطي نغرسه في مشكلة VRP، وأخيراً، نبحث عن الموضع المثالي لRMA . على وجه الخصوص، نقارن الحلول الناتجة عن خوارزمياتنا الثلاث من أجل تحديد علم التمثيل الغذائي الأنسب لهذه المشكلة.

الكلمات الرئيسية: البحوث التشغيلية، الجدولة، اللوجستيات الحضرية، النقل، VRP، meta-heuristics، الخوارزميات الجينية،hybrid meta-heuristics ،simulated annealing.