UNIVERSITE MOHAMED BOUDIAF DE M'SILA
Faculté des Mathématiques et de l'Informatique
Département de Mathématiques

Présenté pour l'obtention du diplôme de MASTER

Domaine : Mathématiques et de L'informatique

Filière : Mathématiques

Option : Analyse Mathématique et Numérique

Présenté par
BEN SERAI NOR EL HOUDA

Sujet

Efficacité et fiabilité des algorithmes méta-heuristiques pour un problème d'ordonnancement

Soutenu le :27/06/2022

Devant le jury :

Mr. SELT Omar	Prof, Univ de M'sila	Rapporteur
Mr. DILMI Mustapha	M.C.B, Univ de M'sila	Président
Mr. GAGUI Bachir	M.C.A, Univ de M'sila	Examinateur

Promotion : 2021/2022

Remerciements

Avant tout, j'adresse mes remerciements en premier lieu, à ALLAH puissant pour la volonté, le courage et la patience qu'il m'a donné durant toutes ces longues années de formation.

Je tiens à remercier sincèrement ***Mr : SELT Omar****, pour avoir accepté de diriger ce mémoire, pour ses conseils.*

Mes remerciements vont égallement aux membres du jury qui m'ont fait l'honneur d'examiner ce travail.

Enfin je ne voudrais pas oublier de remercier toute personne qui m'a aidé à réaliser ce travail.

Merci

Dédicace

Je dédie ce modeste travail :

-A mes parents Ma bougie de ma vie ma mère et à mon père,

-Au défunt qui reste dans mon cœur mon cher grand-père,

-A mes frères : Mouhamad abd allah,Ma prince Siraj el dinne,

-A mes sœurs : Douaa,Arwa,Hibet el rahmmen,Ma princesse Abrar,

-A toute la famille Ben serai et Belaribi,

-A tous mes amis et toute ma famille de département

de Mathématiques,

-A toutes mes adorables que j'ai connu pendant

toute ma vie . . .

Ben serai Nor el houda

Table des matières

Notation

Nous introduisons les notations et les définitions nécessaires qui sont utilisées par la suite.

S A Simulated Annealing

T S Tboue Search

I L S Iterated Local Search

A C O Ante Colonie Optimisation

E A Evolutionary Algorithms

Introduction générale

Les mathématiques sont une science très large qui contient de nombreuses difficultés et concepts et qui développe encore à ce jour, et parmi les concepts que les scientifiques étudient encore L'ordonnancement et ses problèmes.

Les problèmes d'ordonnancement formulés en problèmes d'optimisation sont souvent classées NP –Difficiles, en particulier ceux liés aux systèmes de production la résolution de tels problèmes nécessitent des méthodes dédiées, tandis que les méthodes exactes ne peuvent pas résoudre ce types de problème vu le temps de calcul énorme les méthodes approchées offrent la possibilité de trouver une solution réalisable en un temps raisonnable.

Parmi les méthodes approchées les plus utilisées pour la résolution des problèmes NP-Difficiles on trouve les méta-heuristiques. Durant des années plusieurs méta heuristiques ont été construits et appliquées pour la résolution de tels problèmes.

Mon travail est composé de 3 chapitres :

Le premier chapitre est consacré la présentation des notions générales liées à mon travail, j'ai apprendre ce que cela signifie L'ordonnancement et leur fonctions techniques de l'industrie et on va expliquer les données d'un problème d'ordonnancement et leur classification.

Le deuxième chapitre a pour but d'introduire les méthodes de résolutions des problèmes L'ordonnancement, commençant par les méthodes approchés allant aux heuristiques et les métaheuristiques, qui comprende un ensemble d'algorithmes permettant de trouver la solution la plus rapide et la plus efficace pour une large gamme de problèmes d'optimisation difficile et pour lesquels on ne connaît pas de méthode classique plus efficace.

Le troisième chapitre est une présentation la méthode de recherch tabou et en déduire les avantages et les inconvénients de l'algorithme.

Chapitre 1

Génnéralité sur l'ordonnancement

1.1 Introduction

Un problème d'ordonnancement consiste à organiser dans le temps la réalisation de tâches ou les opérations, compte tenu d'un certain nombre de contraintes afin d'atteindre un certain objectif appelé fonction économique.

L'ordonnancement joue un rôle essentiel dans de nombreux secteurs d'activités : la conception (de bâtiments, de produits, de systèmes, . . .), l'administration (gestion d'emplois du temps, gestion du personnel), l'industrie (gestion de la production), l'informatique (ordonnancement de processus, ordonnancement de réseaux). Les méthodes d'ordonnancement foisonnent dans la littérature. Elles se différencient par la nature du problème considéré (nombre de ressources, structure particulière du problème, . . .), la nature des contraintes prises en compte, les objectifs à satisfaire (minimisation des coûts, de la durée totale de mise en œuvre,. . .) et la nature de l'approche de résolution adoptée (heuristiques, méthodes exactes, métaheuristiques, approches par contraintes, . . .).

La maîtrise de l'ordonnancement est d'un intérêt primordial pour les entreprises qui sont confrontées à des contraintes de productivité, de réactivité et de flexibilité. L'étude de l'ordonnancement est également d'un intérêt théorique toujours renouvelé pour les chercheurs puisqu'il n'existe pas encore de méthodes à la fois générale et de faible complexité algorithmique à cause de la nature fortement combinatoire des problèmes d'ordonnancement.

1.2 Défnition

Un problème d'ordonancement consiste à organiser dans le temps la rèalisation de tàches,compte tenu de contraintes temporèlle (dèlais,contraintes d'enchainement) et de contrainte portant sur la disponabilité des ressources requses.

L'ordonancement appartient à trois fonctions techniques de l'industrie.

Etude

Production

Maintenance

1.2.1 L'Ordonancement dans l'etude

Un Ordonancement constitue une solution au probléme d Ordonancement il est définir par le planning d exécution des tàches (ordre et calendrier) et d allocation des ressources et vise à satisfaire un ou plusieurs objectifs.

1.2.2 L'Ordonancement dans la production

En production (manufacturiére, de biens , de service), on peut le présenter comme un probléme ou'il faut réaliser le déclenchement et le controle de l'avancement d'un ensemble de commandes à traverse les différents centres composant le systéme.

1.2.3 L'Ordonancement dans la maintenance

Systeme de communication rélatif à une intervention corrective , entre le moment d'apparition et la remise a niveau de l'equipement défaillant pour pouvoir fonctionner correctement le service d'Ordonancement a besoins d'informations provenant du bureau des méthodes ces principale.

1.3 Mission de l'ordonancement

Prévoir la chronologie du déroulement des tàches , optimiser l'utilisation des moyens nécessaires, et les rendre disponibles, lancer les travaux au mement choisi, controler l'avancement el la fin des taches, et prendre en comptr les écartes entre prévisions et réalisations.

1.4 Les trois étapes de l'ordonancement

1.4.1 La planifcation

Qui vise à déterminer les différents operations à réaliser, les dates correspondantes , et les moyens materiels et humains à Yeffecter.

1.4.2 L'exécution

Qui consiste à la mise en oeure des différentes opérations défnis dans la phase de planifcation.

1.4.3 Le controle

Qui consiste à effectuer une comparaison entre planifcation et excution, soit au niveau des couts , soit au niveau des dates de réalisation.

1.5 Analyse de temps en maintennance

1.5.1 Préparation

C'est le délais de sous traitance, budget, état.

1.5.2 Ordonnancement

Planning charge repartition, de travail le temps.

1.5.3 Intervention

Le temps passé est un delais à réspecter.

1.6 Classifcation des problèmes d'ordonnancement

Un des paramétres primodiaux pour la génération d'un ordonnancement optimale dans les problémes d'ordonnancement job shop, est l'exploitation maximale du temps, car un simple décalage des opérations retardées peut éviter les trous de temps (temps morts) afin de préparer un or donnancement (compact).

En effet, plus un ordonnancement est compact, meilleure est sa qualité cette notion de compacité qui est un objectif majeur de tout ordonnancement est la base de distinction de plusieurs classes d'ordonnancement : admissible, semi-actif, actif, sans délai.

1.6.1 Ordonnancement admissible

Si toutes les contraintes du probleme sont bien respectées, l'ordonnancement est dite : (admissible) dans certains cas, des décalage à gauche sur certaines opérations sont nécessaires.

Selon que l'ordre des opérations reste inchangé ou non, nous distinguons deux cas :

1 décalage gauche locale : l'anvancement du début d'une opération ne remet pas en cause l'ordre des autre opérations.

2 décalage gauche globale : l'anvancement du début d'une opération engendre une modiffcation au niveau de l'ordre relatif aux deuxopérations au minimum.

1.6.2 Ordonnancement semi-actif

Si aucun décalage locale n'est possible, l'ordonnancement est dit semi- actif donc aucune opération ne peut ètre exécutée en plus tot sans modiffer l'ordre relatif au moins de deux opérations.

1.6.3 Ordonnancement actif

Si aucun décalage à gauche que se soit local ou global n'est pas possible, l'ordonnancement est dit actif en concéquence il est impossible d'avancer une opération sans reporter le début d'une autre opération.

1.6.4 Ordonnancement sans délais

Un ordonnancement est dit sans délai ou retard, si et selement si aucune opération n'est mise en attente alors qu'une machine est disponible pour l'exécuter.

A noter que la transformation en un ordonnacement sans délai peut mener à une solution plus mauvaise du point de vue makespan il existe une relation d'inclution entre les différentes classes d'ordonnancement précédentes.

1.7 Notion de complexitéde problèmes

La théorie de la complexité a pour but d'apporter des informations sur la diffculté théorique d'un problème à résoudre. Elle permet de classer "du point de vue mathématique" les problèmes selon leur diffculté.

La complexité analyse le temps nécessaire pour obtenir une solution (on peut également s'intéresserà la mémoire nécessaire, mais nous ne nous intéresserons pas ici à cet aspect de la complexié). On peut considérer la durée moyenne ou la durée dans le pire des cas. Nous traitons essentiellement ce dernier cas.

A premiére vue, comment définir un algorithme efficace Pour un problème donné ? chercher un algorithme efficace, veut dire trouver un algorithme ou le temps nécessaire à son exécution ne soit pas trop important. Un problème est dit facile si on peut le résoudre facilement, c'est-à-dire s'il ne fait pas trop de temps pour arriver à la solution.Donc, s'il existe un algorithme efficace pour un problè me donné, alors ce dernier est dit facile. Un problème pour lequel on ne connaît pas d'algorithme efficace, alors ce dernier est dit difficile Pour résoudre un problème d'ordonnancement, il ne suffit pas de prouver l'existence d'une solution, il faut également la construire, il est clair que construire la solution est plus difficile que de prouver son existence ce qui nous conduit donc à classer les problèmes comme étant difficiles ou faciles.

Les problèmes indécidables sont ceux pour lesquels aucun algorithme, quel quíil soit, n'a été trouvé pour les résoudre.A l'opposé, les problèmes décidables sont ceux pour lesquels il existe au moins un algorithme pour les résoudre.

1.7.1 Les Classes P et NP

La classe NP est celle des problèmes d'existence dont une proposition de solution est Oui et qui est vérifiable polynomialement. Parmi les problèmes décidables, les plus simples à résoudre sont regroupés dans la classe NP. La classe NP est également décomposée en trois catégories qui permettent d'identifier les problèmes les plus simples et les problèmes les plus compliqués de la classe. Un problème est dit polynomial s'il existe un algorithme de complexité polynomiale permettant de répondre à la question posée dans ce problème, quelle que soit la donnée de celui ci. La classe P est l'ensemble de tous les problèmes de reconnaissance polynomiaux Pour le reste de la classe NP, on n'est pas sûr qu'il n'existe pas un algorithme polynomial pour résoudre chacun de ses problèmes. Ainsi, on sait que P est inclue dans NP mais on n'a pas pu prouver que P n'est pas NP.

1.7.2 La classe NP-Complet et NP-Difficile

La classe NP-Complet regroupe les problèmes les plus difficiles de la classe NP. Elle contient les problèmes de la classe NP tels que n'importe quel problème de la classe NP leur est polynomialement réductible. Entre eux, les problèmes de la classe NP-Complet sont aussi difficiles. La classe NP-Difficile regroupe les problèmes (pas forcément dans la classe NP) tels que n'importe quel problème de la classe NP leur est polynomialement réductible.

1.8 Problème d'optimisation

Un problème d'optimisation se définit comme la recherche, parmi un ensemble de solution possibles S (appelé aussi espace de décision ou espace de recherche), la (ou des solution(s)) x^*qui rend(ent) minimale (ou maximale) une fonction mesurant la qualité de cette solution. Cette fonction est appelée fonction objectif ou fonction coût. Si l'on

pose $f : S \to \mathbb{R}$ la fonction objectif à minimiser (respectivement à maximiser) à valeurs dans $\mathbb{R}$, le problème revient alors à trouver l'optimum $x^* \in S$ tel que $f(x)$ soit minimal (respectivement maximal).

Lorsque l'on veut résoudre un problème d'optimisation, on recherche la meilleure solution possible à ce problème, c'est-à-dire l'optimum global. Cependant, il peut exister des solutions intermédiaires, qui sont également des optimums, mais uniquement pour un

sous-espace restreintde l'espace de recherche : on parle alors d'optimums locaux. La seule hypothèse faite sur s est qu'il s'agit d'un espace topologique, i.e.sur lequel est définie

une notion de voisinage. Cette hypothése est nécessaire pour définir la notion de solution locale du problème d'optimisation. On peut alors définir un optimum local (relativement au voisinage V) comme la solution x^* de S telle que $f\ (x^*) \leqslant\ f(x); \forall x \in V(x^*)$

1.9 Conclusion

L'ordonnancement est généralement décrit comme une fonction particulière de décision au sein d'un système de gestion du travail concernant la production de bien,d'ouvrages ou de services.

La majorité des problèmes d'ordonnancement sont NP-difficile, ça veut dire que, dans la pratique, la complexité croût exponentiellement . Il n'est, donc, pas envisageable de résoudre de tels problèmes avec les méthodes exactes. C'est pour cela qu'il faut développer des nouveaux méthodes qui donnent des solutions certes sous-optimales, maisobtenues rapidement. dont l'objectif est de fournir des solutions aussi proches que possible de la solution exacte en un temps raisonnable.

Chapitre 2

Méthodes de résolution des problémes d'ordonnancement

2.1 Introduction

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

2.2 Les méthodes exactes

Elles recherchent un ordonnancement optimal qui minimise ou maximise un des critères présentés ou une combinaison de plusieurs critères. Les techniques utilisées sont les méthodes par séparation et évaluation, la programmation dynamique, la déduction mathématique, la théorie des jeux, la théorie des graphes etc. Cette technique est coûteuse en temps de calcul.[5]

2.3 Les méthodes approchées

Elles recherchent une solution réalisable de la fonction objectif, mais sans garantir d'optimalité. Elles sont plus pratiques pour la résolution de problèmes difficiles ou NPdifficiles.

Ces méthodes sont souvent classées en deux catégories : des méthodes heuristiques et des méthodes méta-heuristiques.

2.3.1 Les méthodes heuristiques

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

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

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

Définition : Les heuristiques sont des ensembles de règles empiriques ou des stratégies qui fonctionnent, en effet, comme des règles d'estimation.

Exemples d'heuristiques :

► **FIFO** (First In First Out) : la première tâche qui vient est la première tâche ordonnancée.

► **SPT** (Shortest Processing Time) : la tâche ayant le temps opératoire le plus court est traitée en premier lieu.

► **LPT** (Longest Processing Time) : la tâche ayant le temps opératoire le plus important est ordonnancée en premier lieu.

► **EDD** (Earliest Due Date) : cet algorithme choisit parmi les tâches exécutables celle dont le délai est échu le plus tôt. Si aucune tâche n'est disponible, alors un temps libre est généré.

► **SRPT** (Shortest Remaining Processing Time) : cette règle, servant à lancer la tâche ayant la plus courte durée de travail restant à exécuter, est très utilisée pour minimiser les encours et dans le cas des problèmes d'ordonnancement préemptifs.

► **ST** (Slack Time) : à chaque point de décision, l'opération ayant la plus petite margetemporelle est prioritaire. Faute de disponibilité des ressources de production, cette

marge peut devenir négative.

2.3.2 Les méthodes méta-heuristiques

Le mot méta-heuristique est composé de deux mots ; le mot méta qui est un préfixe signifiant "au-delà" ou bien "dans un niveau supérieur" ; et le mot heuristique qui vient du verbe heuriskein et qui signifie ‘trouver’.[4]

On cite quelques définitions de la notion ‘ méta-heuristiques’ :

Définition : Une métaheuristique est formellement définie comme un processus de génération itératif qui guide une heuristique subordonnée en combinant intelligemment différents concepts pour explorer et exploiter l’espace de recherche, des stratégies d’apprentissage sont utilisées pour structurer l’information afin de trouver efficacement des solutions quasi optimales .[4]

Définition : Une métaheuristique est un ensemble de concepts qui peuvent être utilisés pour définir des méthodes heuristiques pouvant être appliquées à un large éventail de problèmes différents. En d’autres termes, une métaheuristique peut être considérée comme un cadre algorithmique général qui peut être appliqué à différents problèmes d’optimisation avec relativement peu de modifications pour les adapter à un problème spécifique. Des exemples de métaheuristiques incluent le recuit simulé (SA), la recherche tabou (TS), la recherche locale itérée (ILS), les algorithmes évolutionnaires (EA) et l’optimisation des colonies de fourmis (ACO) .[4]

Les propriétés fondamentales des métaheuristiques :

⋆ Les métaheuristiques sont des stratégies qui permettent de guider la recherche d’une solution optimale.

⋆ Le but visé par les métaheuristiques est d’explorer l’espace de recherche efficacement afin de déterminer dessolutions (presque) optimales.

⋆ Les techniques qui constituent des algorithmes de type métaheuristique vont de la simple procédure derecherche locale à des processus d’apprentissage complexes.

⋆ Les mtaheuristiques sont en général non-déterministes et ne donnent aucune ga rantie d’optimalité.

⋆ Les métaheuristiques peuvent contenir des mécanismes qui permettent d’éviter d’être bloqué dans des régionsde l’espace de recherche.

⋆Les concepts de base des métaheuristiques peuvent être décrit de manière abstraite, sans faire appel à un problème spécifique.

⋆Les métaheuristiques peuvent faire appel à des heuristiques qui tiennent compte de la spécificité du problèmetraité, mais ces heuristiques sont contrôlées par une stratégie de niveau supérieur.

⋆ Les métaheuristiques peuvent faire usage de l'expérience accumulée durant la re cherche de l'optimum, pourmieux guider la suite du processus de recherche.

Classification des métaheuristiques : On peut regrouper les métaheuristiques en deux grandes classes : les métaheuristiques à solution unique (c.à.d. évoluant avec une seule solution) et celles à solutions multiples ou population de solutions. Les méthodes d'optimisation à population de solutions améliorent, au fûr et à mesure des itérations, une population de solutions. L'intérêt de ces méthodes est d'utiliser la population comme facteur de diversité.

Les métaheuristiques à solution unique :

Dans cette section, nous présentons les métaheuristiques à base de solution unique, aussi appelées méthodes de trajectoire. Contrairement aux métaheuristiques à base de population, les métaheuristiques à solution unique commencent avec une seule solution initiale et s'en éloignent progressivement, en construisant une trajectoire dans l'espace de recherche.

Les méthodes de trajectoire englobent essentiellement la méthode de descente, la méthode du recuit simulé, la recherche tabou, la méthode GRASP, la recherche à voisinage variable, la recherche locale itérée, et leurs variantes.

►Les méthodes de descente (DM : Descent method) :

Appelée aussi méthode d'amélioration itérative ou Hill Climbing en anglais est assez ancienne [3] Sa rapidité et sa simplicité font d'elle la méthode la plus utilisée. Elle part d'une solution initiale X_i d'un ensemble de recherche S et progresse vers une solution voisine X_{i+1} de meilleure qualité quel que soit i. Son algorithme général pour un problème de minimisation est donné par :

Répéter :
1) Choisir X_{i+1} dans $V(X_i)$
2) Si $f(X_{i+1}) < f(X_i)$ alors $X_i \leftarrow X_{i+1}$
3) Jusqu'à ce que $f(X_{i+1}) \geq f(X_i)$, $\forall X_{i+1} \in S$
4) Fin.

Pour appliquer une descente, il faut bien choisir la solution initiale (X_i) qui est généralement aléatoire ou provenue d'une méthode approchée. Quand cette solution est aléatoire, on pourra appliquer plusieurs fois la descente en changeant à chaque fois la solution initiale et ne mémorisant que celle donnant le meilleur résultat. La méthode de descente est très facile à programmer. Son inconvénient réside dans le fait qu'elle calcule une solution locale car elle s'arrête au premier optimum rencontré. La répétition de déclaration de la solution initiale permet d'atténuer cet inconvénient.

•Avantages et inconvénient

◦ Avantages

- la plupart du temps, elle ne fait que calculer$f(s+i)-f(s)$, où i correspond à un déplacement élémentaire, et si cette expression peut se simplifier algébriquement, alors on pourra évaluer très rapidement cette différence.

◦ inconvénients

- la recherche s'arrête au premier minimum local rencontré.

►La méthode GRASP :

La procédure de recherche gloutonne aléatoire adaptative (GRASP : Greedy Randomized Adaptive Search Procedure), proposée par Feo et Resende dans [Feo & Resende, 1989, 1995], est une métaheuristique à départs multiples, dépourvue de mémoire, fondée sur les algorithmes gloutons randomisés et les techniques de recherche de voisinage[8]. Chaque itération de l'algorithme GRASP se compose de deux étapes dites de construction et de recherche locale. L'étape de la construction est similaire à l'heuristique semi-gloutonne (semi-greedy heuristic) proposée indépendamment par Hart et Shogan [Hart & Shogan, 1987]. Elle génère une solution réa lisable par l'application d'une procédure d'initialisation gloutonne. Dans la deuxième étape, cette solution est utilisée comme solution initiale de la procédure de recherche locale. Après un nombre donné d'itérations, l'algorithme GRASP se termine et la meilleure solution trouvée est conservée. Une étude bibliographique de la méthode GRASP est fournie dans [Festa & Resende, 2009a,b].L'algorithme de GRASP pour un problème de minimisation est le suivant :

T = dimension du problème ;
Pour $k = 1$ jusqu'à itération-max faire
Construction :
Solution = 0 ;
Evaluer le cout incrémental des éléments candidats ;
Tant que Dimension ($solution$) $\neq T$: Construire LCR : la liste de candidats restreinte ;
Sélectionner au hasard une solution s de LCR : solution = Ajouter ($solution, s$)
Fin
Recherche locale :
Tant que solution non optimale localement :
Trouver $s' \in Voisinage(solution)$ tel que $f(s') < f(solution)$
Solution = s'
Fin
meilleure-solution = min (solution, meilleure-solution) ;
Fin
Retourner meilleure-solution

• **Avantages et inconvénients**

◦ **Avantages**

- La méthode GRASP est relativement simple à programmer.

- Elle ne nécessite que peu de paramétrage : la taille de la liste, qui permet d'équilibrer la quantité d'adaptabilité (l'heuristique) et le facteur stochastique.

- Elle nécessite de plus peu de calculs supplémentaires par rapport à une recherche locale simple.

◦ **Inconvénients**

- Elle est en revanche moins performante que d'autres métaheuristiques, essentiellement du faitde son absence de mémoire.

-Rien ne garantit que l'algorithme ne va pas, à plusieurs reprises.

-Explorer des zones très similaires et retomber sur les mêmes minima locaux.

Les métaheuristiques à population de solutions :

Contrairement aux algorithmes partant d'une solution singulière, les métaheuristiques à population de solutions améliorent, au fur et à mesure des itérations, une population de solutions. On distingue dans cette catégorie, les algorithmes évolutionnaires, qui sont une famille d'algorithmes is sus de la théorie de l'évolution par la sélection naturelle, énoncée par Charles Darwin et les algorithmes d'intelligence en essaim qui, de la même manièreque les algorithmes évolutionnaires, proviennent d'analogies avec des phénomènes biologiques-naturels.

►**Les algorithmes génétiques (GA :Genetic Algorithm) :** Proposé dans les années 1975 par Holland, les algorithmes génétiques doivent leur popularité à Goldberg. Avant la parution de son livre qui est une des références les plus citées dans le domaine de l'informatique, on a puvoir un certain nombre d'autres présentations, citons Goldberg, Holland, Schwefel Le sujet connaît une très grande popularité. Il existe aujourd'hui plusieurs milliers de références sur le sujet et le nombre de conférences dédiéesau domaine (que ce soit sur les techniques elles-mêmes ou sur les applications)ne fait qu'augmenter[10].

De manière générale, les algorithmes génétiques utilisent un même principe.Une population d'individus (correspondants à des solutions) évoluent en mÍme temps comme dans l'évolution naturelle en biologie. Pour chacun des individus, on mesure sa faculté d'adaptation au milieu extérieur par le fitness.

Les algorithmes génétiques s'appuient alors sur trois fonctionnalités :

- La Sélection : Pour déterminer quels individus sont plus enclins à se reproduire, une sélection est opérée. Il existe plusieurs techniques de sélection, les principales utilisées sont la sélection par tirage à la roulette (roulette-wheel selection), la sélection par tournoi (tournament selection), la sélection par rang (ranking selection), etc[11].

- Le Croisement : L'opérateur de croisement combine les caractéristiques d'un en semble d'individus parents (généralement deux) préalablement sélectionnés, et génère de nouveaux individus enfants. Là encore, il existe de nombreux opérateurs de croisement, par exemple le croisement en un point, le croisement en n-points ($n \succeq 2$) et le croisement uniforme[11].

- La Mutation et le remplacement : Les descendants sont mutés, c'est-à-dire que l'on modifie aléatoirement une partie de leur génotype, selon l'opérateur de mutation.Le rem placement (ou sélection des survivants), comme son nom l'indique, remplace certains des parents par certains des descendants. Le plus simple est de prendre les meilleurs indivi dus de la population, en fonction de leurs performances respectives, afin de former une nouvelle population (typiquement de la même taille qu'au début de l'itération)[11].

La représentation des solutions (le codage) est un point critique de la réussite d'un algorithme génétique. Il faut bien sûr qu'il s'adapte le mieux possible au problème et à l'évaluation d'une solution. Le codage phénotypique ou codage direct correspond en gén éral à une représentation de la solution très proche de la r éalité. L'évaluation d'une solution représentée ainsi est en général immédiate .Le principe général d'un algorithme génétique est présenté par

1) Initialisation :
K :=nombre total de générations ;
N :=taille de la population ;
Pc :=Probabilité de croisement ;
Pm :=probabilité de mutation ;
POP :=population courante ;
PINT :=population intermédiaire contenant N éléments chacun ;
2) Répéter
Pour $i = 0$ jusqu'à K
Évaluer [POP]
Sélection [PINT, POP]
Croisement [PINT, Pc]
Mutation [PINT, Pm]
3) Remplacement POP=PINT
Evaluer [POP]
Solution meilleur=meilleur [POP]

• **Avantages et inconvénients**

◦ **Avantages**

- Les algorithmes génétiques opérent au niveau du codage des paramètres sans se soucier de leur nature , donc ils s'appliquent à de nombreuses classes de problèmes , qui dépendent éventuellement de plusieurs paramètres de natures différentes(booléens,entiers,réels, fonctions...).

- Pour les mêmes raisons un algorithme génétique est dans l'idéal totalement indépendant de la nature du problème et de la fonctionnelle à optimiser, car il ne se sert que des valeurs d'adaptation, qui peuvent être très différentes des valeurs de la fonction à optimiser, même si elles sont calculées à partir de cette dernière.

- Potentiellement les algorithmes génétiques explorent tous l'espace des points en même temps , ce qui limite les risques de tumber dans des optimums locaux.

- algorithmes génétiques présentent une grande robustesse c'est-à-dire une grande capacité à trouver les optimums globaux des problèmes d'optimisation.

◦ **inconvénients**

- Les algorithmes génétiques ne sont encore actuellement pas très efficaces en coût (ou vitesse de convergence) , vis - à - vis de méthodes d'optimisation plus classiques.

- Parfois les algorithmes génétiques convergent très vite vers un individu particulier de la population dont la valeur d'adaptation est très élevée.

- Le respect de la contrainte de domaine par la solution codée sous forme de chaîne de

bits pose parfois problème . Il faut bien choisir le codage , voire modifier les opérateurs.

- L'utilisation d'un algorithme génétique ne garantie pas le succès de l'optimisation.

- En pratique l'efficacité d'un AG dépend souvent de la nature du problème d'optimisation . Selon les cas de choix des opérateurs et des paramètres seront souvent critiques , mais aucune théorie générale ne permet de connaître avec certitude la bonne paramétrisation , il faudra faire plusieurs expériences pour s'en approcher [9].

► Algorithme de colonies de fourmis (ACO : Ant Colony Optimisation) :

Est un algorithme d'intelligence en essaim dont le principe est basé sur la manière dont les fourmis cherchent leurs nourritures et retrouvent leur chemin pour retourner dans la fourmilière. Initialement, les fourmis explorent les environs de leur nid de manière aléatoire [6] Sitôt qu'une source de nourriture est repérée par une fourmi, son intérêt est évalué (quantité et qualité) et la fourmi ramène un peu de nourriture au nid. Les fourmis peuvent déposer des phéromones au sol, grâce à une glande située dans leur abdomen et former, ainsi, des pistes odorantes qui pourront être suivies par leurs congénères. Les traces laissées s'accumulent au fur et à mesure que la piste est rejointe par plus de congénères. Les phéromones ont comme caractéristique l'évaporation en fonction du temps. Les pistes les plus longues seront donc abandonnées au profit de la plus courte. Ainsi, pour illustrer le principe de cette méthode pose A un ensemble de K fourmis. Une fourmi de cet ensemble va démarrer de la ville i à l'instant t vers la ville j, la destination est choisie en fonction de la visibilité η_{ij} (l'inverse de la distance) et de la quantité de phéromones $\tau_{ij}(t)$ déposée entre les deux villes

---

Répéter :
Pour $i = 1$ à k faire
Construire le Trajet (i)
Fin Pour
Mettre à Jour Phéromones O
Jusqu'à ce que le critère de terminaison soit satisfait.
Dans la procédure Construire le Trajet (i), chaque
fourmi construit un trajet selon la probabilité
$p^k ij(t)$ telque :
$p * ij(t) = [(\tau_{ij}(t))]^{\alpha *}(\eta_{ij})^{\beta} \mid_{sinon\ p^k ij(t)=0} [\sum_{i \in jik}(\tau_{ij}(t))^{\alpha *}(\eta_{\alpha})^{\beta}] sij \in jik$

α et β sont deux paramètres contrôlant respectivement l'influence du taux de phéromone sur le trajet (ij) et l'influence de la distance sur le trajet (ij). La procédure mettre à jour Phéromone O consiste à mettre à jour le taux de phéromones (ij) pour la fourmi k en appliquant La formule suivante :

$\tau_{ij}(t+1) = (1-p) * \tau_{ij}(t) + \Delta\tau_{ij}(t) + \Delta\tau_{ij}(t)$

Avec $\Delta\tau_{ij}(t) = \sum_{x=1}^{k}(\Delta\tau_{ij}^{x}(t))$ et $\Delta\tau_{ij}^{k}(t) = \frac{Q}{LK(T)}$

Q est une constante et $Lk(t)$ est la longueur totale de la tournée de la fourmi k à l'instant t. Pour éviter d'être piégé dans des optima locaux, les algorithmes de colonies de fourmis utilisent la stratégie dévaporation des pistes de phéromones dont la solution est mauvaise [7]. Les algorithmes de colonies de fourmis sont très efficients en optimisation combinatoire grâce à leur robustesse i.e.La recherche reste efficace même si certains de ses individus sont défaillants, et à sa décentralisation, ie. les fourmis n'obéissent pas à une autorité centralisée, comme ils ont l'avantage d'être exécutés en parallèle. Néanmoins, cette méthode présente l'inconvénient de réglage d'un nombre important de paramètres (nombre de fourmis, visibilité, quantité de phéromones etc.).

• **Avantages et inconvénients**

◦ **Avantages**

- Très grande adaptabilité.
- Parfait pour les problèmes basés sur des graphes.

◦ **inconvénients**

- Un état bloquant peut arriver.
- Temps d'exécution parfois long.
- Ne s'applique pas à tous type de problèmes.

2.4 conclusion :

Dans ce chapitre, nous avons présenté quelques méthodes d'optimisation combinatoires en s'appuyant sur les caractéristiques principales des métaheuristiques. Ces dernières sont très efficaces en optimisation difficile sans avoir besoin de modifier la structure de base de l'algorithme utilisé. Elles sont devenues très populaires grâce à leur simplicité d'emploi dans différents domaines. Il est à noter qu'une bonne performance nécessite souvent une formalisation adéquate du problème posé et une adaptation intelligente d'une métaheuristiques. Malgré le succès remarquable de leur démarche, les métaheuristiques présentent des difficultés à lesquelles est confronté l'utilisateur dans le cas d'un problème concret comme le choix d'une méthode efficace pour avoir une solution optimale et le réglage des paramètres qui peut être réalisable en théorie mais irréalisable en pratique. Les chercheurs visent à surpasser ces difficultés en proposant des techniques d'amélioration dont on cite l'hybridation

des métaheuristiques. Cette hybridation exploite la puissance de plusieurs algorithmes, et les combine en un seul méta-algorithme. Le chapitre suivant sera consacré à l'étude des métaheuristiques recherche tabou.

Chapitre 3

La méthode de recherche tabou

3.1 introduction

Dans certains problèmes d'optimisation, les méthodes de résolutions dites exactes, ne permettent pas trouver la solution optimale dans une durée de temps raisonnable.

C'est l'une des principales raisons qui ont contribué à la naissance des métaheuristiques.

En effet cette famille d'algorithmes permet de résoudre des problèmes d'optimisations complexes face auxquels les méthodes classiques manquent d'efficacité. Cependant ces algorithmes de recherche ne peuvent garantir l'optimalité de la solution trouvée. Dans cette section, nous présentons la métaheuristique dont nous nous servirons pour optimiser la gestion des campagnes publicitaires.

En 1986 Fred Glover [1] dévoile pour la première fois sa métaheuristique qu'il baptisa la Recherche Tabou (RT). Cette appellation est tout à fait significative puisque cette méthode, se basant sur une recherche itérative qualifiée de recherche locale au sens large, interdit pour un laps de temps donné de revisiter une solution déjà visitée.

3.2 Méthode de Recherche Tabou (Tabu Search)

La meilleure analogie sans doute pour expliquer l'idée derrière l'algorithme RT serait la fable des randonneurs[2] . Imaginons un randonneur malchanceux perdu dans la montagne. Il voudrait rejoindre le point de plus basse altitude puisqu'il sait qu'une équipe de secours passe régulièrement par ce point. Il ne sait pas qu'elle est l'altitude de son objectif et un brouillard l'empêche de voir loin. Face à un tel problème, la méthodologie de la recherche Tabou lui permettrait d'atteindre son objectif. Partant de n'importe quelle position du randonneur, la RT lui proposera de se déplacer vers le point qui le mènera à la plus basse altitude que le brouillard lui permette de voir. Une fois arrivé, il devra chercher de nouveau parmi toutes les positions qu'il peut apercevoir, la prochaine plus basse position pourvu

qu'il ne l'ait pas déjà visitée. Il arrive dans certains cas que la position choisie soit de plus haute altitude que la position actuelle du randonneur. Le fait de remonter à cette nouvelle position, peut s'avérer une bonne décision puisque cette action peut mener le randonneur à de nouvelles positions encore plus basses que celle qu'il a trouvé depuis son départ. En poursuivant ainsi de suite, le randonneur se déplacera d'un point à un autre tout en mémorisant les T dernières positions qui lui seront taboues.

Il est vrai que la fable des randonneurs[2] favorise la compréhension de l'algorithme Tabou. Mais dans le cadre d'un travail de recherche, il est impératif de passer à une description formelle et rigoureuse de cet algorithme. Dans cette optique, on s'est inspiré de la description d'Alain Hertz de l'algorithme Tabou qu'on présente comme suit :

3.3 Définition du problème

Soit N l'ensemble de toutes les solutions possibles, et F une fonction à optimiser et qui détermine la valeur $F(S)$ de toute solution S dans N.

Le problème à résoudre est donc le suivant :

$$\begin{cases} max\ F(S), \\ \text{sous contraintes : } S \in \mathbb{N}. \end{cases}$$

On appelle voisinage, la fonction V qui associe un sous-ensemble de N à toute solution $S \in N$. Ainsi un voisin de S est toute solution $S\prime \in V(S)$.

Une solution $S \in N$ est considérée comme étant un maximum local dans un voisinage V si :

$$F(S') \leqslant F(S) \forall\ S' \in V(S).$$

Une solution $S \in N$ est dite un maximum global si :

$$F(S') \leq\ F(S),\ \forall\ S'\ \in N.$$

3.3.1 Principe

Durant l'évolution itérative de la recherche Tabou, cet algorithme choisit à chaque itération la meilleure solution $S' \in V(S)$, même si $F(S) > F(S')$. Quand la recherche atteint un maximum local S dans un voisinage V, l'algorithme Tabou sera contraint de se déplacer vers une solution S' avec $F(S) > F(S')$. Dans certain cas, il arrive que le voisinage de la solution $S\prime$ contienne la solution $S \in V(S')$.Alors, même si la solution S est un maximum local dans le voisinage V, il ne faut surtout pas revenir immédiatement à S, sinon la

recherche se trouverait piégée à tourner en rond entre Set S'. Afin d'éviter ce problème, l'algorithme Tabou fait intervenir une liste Tbaptisée liste taboue. Cette liste permet de mémoriser durant un nombre limité d'itérations (mémoire à court terme), les dernières solutions visitées et d'interdire tout déplacement vers ces solutions. On appelle toute solution figurant dans la liste T une solution taboue.

3.3.2 Mémoire

Elle préserve un nombre d'états visités précédemment accompagné d'un nombre d'états qui pourraient être non acceptés.[12]

3.3.3 Tabous et liste tabou

Les taboues sont une manière de représenter la mémoire du cheminement effectué pour diriger l'exploration vers des régions non visitées. La manière la plus simple de définir les tabous est de conserver une liste Taboue qui contient les dernières solutions rencontrées et on empêche à la procédure d'y retourner. On gère cette liste comme une liste circulaire : on élimine le plus vieux tabou et on insère la nouvelle solution (cette solution peut s'avérer coûteuse en terme de quantité d'information requise).

Le rôle de la liste tabou est d'interdire les mouvements cycliques. La longueur de la liste doit être bien choisie. La valeur moyenne des solutions visitées se développe proportionnellement avec l'augmentation de la taille de liste taboue .

3.4 Algorithme de la recherche Tabou

- S : Solution courante ;
- S^* :Meilleure solution rencontrée depuis le début de la recherche

1) Poser $T \leftarrow \emptyset$ { initialiser une liste taboue vide}
2) Générer une solution initiale s.
3) Metter $s^* \leftarrow s$ { s^* est la meilleure solution }
4) Répéter
5) Générer $N(s)$
6) Choisir s' qui minimise $f(s')$ dans $N(s)$
7) Si $f(s') < f(s^*)$ alors
8) Poser $s^* \leftarrow s'$ { condition d'aspiration : si la solution courante améliore la meilleure solution ; accepter la méme si elle est taboue }
9) Poser $s \leftarrow s'$ { metter à jour la solution courante }
10) $T \leftarrow T + s'$ { metter à jour la liste tabou }
11) sinon si $s' \in NT(s)$ alors
12) Poser $s \leftarrow s'$ { metter à jour la solution courante si elle n'est pas taboue }
13) Si $f(s') < f(s^*)$ alors
14) Poser $s^* \leftarrow s'$
15) $T \leftarrow T = s'$
Fin.

À part le fait que l'algorithme Tabou a besoin de mémoriser la liste taboue pour éviter de cycler, la RT mémorise aussi tout au long de son évolution la meilleure solution S^*rencontrée. En effet, à la fin de la recherche, lorsque le critère d'arrêt est vé rifié, S^* nous permettra de retrouver la meilleure solution visitée depuis le lancement de la recherche.

Par ailleurs, l'analyse de l'algorithme présenté dans cette section nous permet de déceler rapidement trois critères dont la configuration influe considérablement sur les résultats de la recherche en qualité et en temps. Le premier étant la taille de la liste taboue. Malgré l'apport positif que fournit cette liste, un mauvais paramétrage de sa longueur risque de lui faire perdre toute son utilité. D'un côté, plus la taille de cette liste est petite, plus le phénomène de cyclage reprend vie. D'un autre côté, une liste taboue trop longue limitera énormément l'espace de recherche. Face à ce dilemme, il est généralement conseillé de procéder à une étude empirique dans l'optique de fixer la bonne taille de la liste taboue.

Le second élément à définir est la fonction$V$ qui détermine le voisinage. La détermination de cette fonction a une grande influence sur l'évolution de la recherche. En effet, une mauvaise définition du voisinage pourrait ralentir considérablement l'évolution de l'algorithme. Donc la question à laquelle il faudra répondre pour définir le voisinage est : partant d'une solution S, comment choisir le sous ensemble des solutions $S' \in V(S)$?

Finalement le troisième élément dont la configuration est assez fragile, est la condition d'arrêt. Ce dernier joue un rôle essentiel du point de vue de la qualité de la solution et

du point de vue de son temps de réponse. On se retrouve ici face à un second dilemme : une condition d'arrêt rapidement atteignable ne laissera pas assez de temps au Tabou pour trouver l'optimum global recherché. À l'inverse, un critère d'arrêt peu probable prolonge trop longtemps la recherche même si l'optimum absolu est atteint.

Donc, afin de répondre aux besoins des annonceurs sur Internet, le module d'optimisation développé au cours de ce travail devra être capable de trouver une bonne solution dans un délai jugé raisonnable. Pour ce faire, nous proposons tout d'abord, d'implémenter l'algorithme de la recherche Tabou puisqu'il a démontré ses preuves dans plusieurs domaines d'application autant difficiles les uns que les autres. Ensuite il faudrait passer par une phase de paramétrage de cet algorithme dont le but est d'améliorer ses performances.

3.5 Techniques d'améliorations

Dans cette section, nous présentons les trois principales techniques permettant d'améliorer les performances et l'efficacité de l'algorithme Tabou.

3.5.1 Critère d'aspiration

Le critère d'aspiration fût introduit la première fois en 1986 par Fred Glover [1]. En 1989, il publie un autre article [13] sur son algorithme Tabou où il explique entre autre le fonctionnement du critère d'aspiration. L'idée derrière cette technique amélioratrice est d'accepter certains mouvements récemment effectués et qui en temps normal, ne seraient pas acceptés par les règles du Tabou. Le fait qu'on gère ici des mouvements plutôt que des solutions, nous permet de mémoriser uniquement ces mouvements au lieu de mémoriser des solutions complètes qui alourdissent considérablement la recherche. Cette problématique est principalement rencontrée lors de la vérification de présence d'une solution voisine dans la liste taboue. Mais la mémorisation des mouvements ne présente pas que des avantages. Le fait de mémoriser des mouvements peut amener à visiter la même solution plusieurs fois et cela sans enfreindre l'interdiction qu'impose la liste taboue. Un autre inconvénient au fait de mémoriser des mouvements est qu'il se peut que la liste taboue ne permette pas de visiter des solutions qui n'ont jamais été explorées, même si celles-ci peuvent être des optimums. Face à ce dernier défaut relatif à la mémorisation des mouvements, l'utilisation du critère d'aspiration est d'une grande utilité. En effet, en libérant de la liste taboue uniquement les mouvements satisfaisant le critère d'aspiration, on peut trouver des solutions meilleures

que la meilleure solution rencontrée depuis le début. Donc, le critère d'aspiration peut décider quand un mouvement est bénéfique et donne ainsi la permission à l'algorithme d'entreprendre ce mouvement même s'il est interdit.

3.5.2 Technique de diversification

Fred Glover nous explique bien cette technique dans son article [6]. En résumé ce processus consiste à effectuer des mouvements visant à varier l'échantillon de recherche. En effet il s'avère intéressant de diversifier les zones de recherches dans l'espoir de couvrir le plus possible de zones dans l'espace N. Généralement, cette technique est appelée lorsque la recherche se trouve bloquée aux alentours d'un optimum local. Grâce à la diversification, une solution sera générée, à partir de laquelle l'algorithme reprendra sa recherche.

3.5.3 Technique d'intensification

Inversement à la technique de diversification, l'intensification effectue des mouvements permettant d'améliorer rapidement la solution. En effet, en favorisant la recherche aux alentours des zones les plus prometteuses, ce processus peut amener à des solutions de meilleure qualité. Comme son nom l'indique, cette technique intensifie et localise plutôt la recherche tout prêt des solutions avantageuses dans l'espoir de les améliorer et d'atteindre des solutions meilleures que S^*.

3.6 Avantages et inconvénients de la recherch tabou

3.6.1 Les avantages

• Offre des économies de temps de resolution pour des programmes de grosse taille.

•L'efficacité de la méthode taboue offre son utilisation dans plusieurs problèmes d'optimisation combinatoire classiques tels que le problème de voyageur de commerce, le problème d'ordonnancement. . . etc.

• Trés bons résultats sur certains types de problèmes NP Difficile.

• Algorithèmes faciles à mettre en oeuver.

3.6.2 Les inconvénients

• Paramètres peu ituitifs.

• Demande en ressources importantes si la liste des tabous est trop imposante.

• la méthode taboue exige une gestion de la mémoire de plus en plus lourde en mettant des stratégies de mémorisation.

• Aucune démonstration de la convergence.

• Aucune garantie optimality.

3.7 Conclusion

Ce chapitre nous a permis d'avoir une vue générale sur les concepts de la recherche taboue qui sont des algorithmes simples de conception et peuvent résoudre des problèmes assez complexes. La résolution de ces problèmes est obtenue grâce aux opérateurs de reproduction.

La Recherche Taboue est une procédure assez robuste pour résoudre les problèmes d'optimisation.

Conclusion générale

Dans ce mémoire, nous intéressées à l'étude de l'ordonnancement en globalité, en rapport les différents méthodes d'optimisation pour trouver de bonnes solutions aux problèmes correspondants en utilisant pour leur résolutions, des méthodes exactes et des méthodes approchées : La méthode GRASP, algorithme de descente DS, les algorithmes génétiques GA, algorithme de colonies de fourmis ACO. A la fin de ce mémoire nous avons présenté un exemple de métode méta-houristique(recharch de tabou). En suite, nous avons évoqué les aventages et les inconvénients de cette méthode.

Bibliographie

[1] F. GLOVER, ″FUTURE PATHS FOR INTEGER PROGRAMMING AND LINKS TO ARTIFICIAL INTELLIGENCE," 1986.

[2] I. Charon, A. Germa et O. Hudry, ″Méthodes approchées définies par un voisinage," in Méthodes d'optimisation combinatoire, Paris : Masson, 1996, pp. 165-185.

[3] C.H.Papadimitriou. The complexity of combinatorial optimization problems. PhD thesis, Princeton, 1976

[4] HANNACHE Aboubakr et LEMMOUIS Abdelhamid. Résolution d'un problème d'ordonnancement de type job shop avec contrainte de transport. Thèse de doctorat. Université de Tlemcen, 2016.

[5] BEN AHMED Razika et KHERROUBI Zakiya. Optimisation d'un problème d'ordonnancement de type job shop avec contrainte de transport. Diss.

[6] Glover, F : Tabu search—Part II, ORSA J. Comput. 2 :4-32. 1990.

[7] Johann Dréo, Patrick Siarry : Métaheuristiques pour l'optimisation et auto- organisation dans les systèmes biologiques. Université de Paris XII Val-de-Marne.

[8] T. A. Feo and M. G. C. Resende. Greedy randomized adaptive search procedures. Journal of Global Optimization, 6(2) : 109–133, 1995.

[9] J. Kennedy , R - C . Eberhart, "Particle swarm optimization," In Proceedings of the IEEE International Conference on Neural Networks, Piscataway Jersey, USA, Vol 5, 1995, p. 1942 .

[10] J. R. Koza. Genetic Programming : On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems). The MIT Press, 1 edition, December 1992. ISBN 0262111705.

[11] J. R. Koza. Introduction to genetic programming. In Kenneth E. Kinnear, Jr., editor, Advances in Genetic Programming, chapter 2, pp. 21–42. MIT Press, Cambridge, MA, USA, 1994.

[12] Johann Dréo — Patrick Siarry, « Métaheuristiques pour l'optimisation et auto organisation dans les systèmes biologiques », (LERISS,A 412), 2004

[13] F. Glover, "Tabu search-part I," ORSA Journal on computing, vol. 1, pp. 190-206, 1989

Abstract

in this work, we prove the scheduling problem in general. With mention of the methods for solving its problems such as the approximate methods and the exact methods,while presenting the advantages and disadvantage of each method. We were the only ones to study the taboo research method.

Keywords : Scheduling,Metaheuristics,Approximate method,Taboo search.

Résumé

Dans ce travail, nous prouvons le problème d'ordonnancement en général. Avec mention les méthodes de résolution de ses problèmes telles que les méthodes approchées et les méthodes exactes,tout en présentant les avantages et les iconvénients de chaque méthode. Nous étions seuls à étudier la méthode de recherch tabou.

Mots clés : L'ordonnancement, Métaheuristique,Méthode approchées, La recherch de tabou .

ملخص

في هذه المذكرة تطرقنا الى مسالة الجدولة بصفة عامة. مع ذكر طرق حل مشاكلها مثل الطرق التقريبية والطرق المضبوطة مع تقديم محاسن ومساوئ كل طريقة. وانفردنا بدراسة طريقة بحث الممنوعات .

الكلمات المفتاحية:الجدولة ، فوقيات الاستدلال ، حل تقريبي ، بحث الممنوعات.