N =° d'ordre : 32/ 2009-M / MT

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ DES SCIENCES ET TECHNOLOGIE DE HOUARI BOUMEDIENNE
FACULTÉ DES MATHÉMATIQUES

MÉMOIRE

Présenté pour l'obtention du Diplôme de Magister.

EN : MATHÉMATIQUE

Spécialité : Recherche Opérationnelle : Mathématiques Discrètes et Optimisation

Par : Saïda Haddadou

Sujet

Approche hybride pour les problèmes d'optimisation combinatoires multi-objectif cas : des enchères combinatoires

Soutenu le 29/09/2009, devant le jury composé de :

Mr D. CHAABANE	Président	Professeur	USTHB
Mr M. AÏDER	Encadreur	Professeur	USTHB
Mr M. MOULAÏ	Examinateur	Professeur	USTHB
Mme C. ADICHE	Invitée	Maitre assitante	USTHB

Table des matières

Remerciements.

Je tiens à exprimer ma profonde gratitude au professeur M. Aïder, d'abord, d'avoir accepté de m'encadrer dans ce travail, puis pour sa disponibilité et ses conseils tout au long du parcours qui nous a réunis.

Je remercie le professeur D. Aissani de l'université de Bejaia pour sa disponibilité, ses 'OUI' pour aider notre club (SCOR).

Je remercie le professeur D. Chaabane pour avoir accepté de présider mon jury.

Mes remerciements vont aussi aux examinateurs, professeur M. Moulaï et Madame C. Adiche pour avoir accepté de juger mon travail et par l'occasion, d'avance, à tous les membres du jury pour leurs critiques et suggestions constructives.

Je remercie mes amis : Aicha, Shara, Kahina, Yassine, Nasredine, Hassene, Abdel-hak pour leurs amitiés et surtout la gentille Nissa de bab-ezzouar.

Dédicaces.

A mes très chers parents, grands parents et mes deux petites soeurs.
A mon essentiel, Moha...

Saïda.

Résumé.

Nous allons traiter dans ce mémoire le problème d'optimisation combinatoire : enchères combinatoires. Ce problème dérive du problème économique des enchères, qui dans le cas le plus élémentaire et le plus banal, est une procédure permettant au propriétaire d'un bien unique et indivisible, qui désire le vendre, de sélectionner l'acquéreur parmi plusieurs candidats (ou, symétriquement, à un acheteur désirant acquérir un objet unique et indivisible de sélectionner le fournisseur parmi plusieurs concurrents).

En optimisation Combinatoire, Les enchères sont des mécanismes où les agents ont la possibilité de lancer des offres non plus simplement sur des objets isolés, mais sur des ensembles d'objets provenant d'une base finie. Elles permettent une expression plus raffinée des préférences des agents et rendent possible une allocation plus efficace des biens.

Le travail réalisé dans ce mémoire porte sur la résolution d'un problème multi-objectif. Les contributions apportées dans ce travail sont : proposition d'une formulation multi-objectif généralisé du problème d'enchères combinatoires et développement d'un algorithme hybride fondé sur l'algorithme de Branch and Bound (exacte) et Recherche Tabou (heuristique).

Abstract.

In this memory, we study the problem of combinatorics optimization : combinatorics auctions. Auctions are important mechanisms for resource and task allocation in multi-agent systems.

In a combinatorial auction, bidders may submit bids on combinations of items. This allows the bidders to express complementarities between items instead of having to speculate into an item's valuation the impact of possibly getting other, complementary items.

In this memory, we study the problem of combinatorial auction and we suggest a more general formulation for this problem which is multi-objective formulation. Then we developed an algorithm based on the hybrid of two methods of optimization Branch and Bound (exact) and Tabu search (heuristic).

Introduction Générale

Résoudre un problème d'optimisation consiste à trouver la ou les meilleures solutions vérifiant un ensemble de contraintes et atteignant des objectifs définis par l'utilisateur. Pour déterminer si une solution est meilleure qu'une autre, il est nécessaire que le problème introduise un critère de comparaison. Ainsi, la meilleure solution, appelée aussi solution optimale, est la solution ayant obtenu la meilleure évaluation au regard du critère considéré. Les problèmes d'optimisation sont utilisés pour modéliser de nombreux problèmes : le traitement d'images, la conception de concret systèmes, la conception d'emplois du temps, etc.
La majorité de ces problèmes sont qualifiés de difficiles, car leur résolution nécessite l'utilisation d'algorithmes évolués, et il n'est en général pas possible de fournir dans tous les cas une solution optimale en un temps raisonnable. Lorsqu'un seul critère est donné, par exemple un critère de minimisation de coût, la solution optimale est clairement définie, c'est celle qui a le coût minimal. Mais dans de nombreuses situations, un seul critère peut être insuffisant. En effet, la plupart des applications traitées intègrent plusieurs critères simultanés, souvent contradictoires.

Intégrer des critères contradictoires pose un réel problème. La solution idéale n'existe pas, et il faut donc trouver un compromis. En effet, en considérant deux critères contradictoires (a) et (b), améliorer (a) détériore forcément (b) et inversement. Le concept de solution optimale devient alors plus difficile à définir. Dans ce cas, la solution optimale cherchée n'est plus un point unique, mais un ensemble de compromis. Résoudre un problème comprenant plusieurs critères, appelé communément problème multi-objectif, consiste donc à calculer le meilleur ensemble de solutions de compromis.

Nous allons traiter dans ce mémoire le problème d'optimisation combinatoire :enchères combinatoires. Ce problème dérive du problème économique des enchères, qui dans le cas le plus élémentaire et le plus banal, est une procédure permettant au propriétaire d'un bien unique et indivisible, qui désire le vendre, de sélectionner

l'acquéreur parmi plusieurs candidats (ou, symétriquement, à un acheteur désirant acquérir un objet unique et indivisible de sélectionner le fournisseur parmi plusieurs concurrents).

En Optimisation Combinatoire, Les enchères sont des mécanismes où les agents ont la possibilité de lancer des offres non plus simplement sur des objets isolés, mais sur des ensembles d'objets provenant d'une base finie. Elles permettent une expression plus raffinée des préférences des agents et rendent possible une allocation plus efficace des biens.

En enchères combinatoires, le vendeur est soumis à un ensemble de coûts donnés pour un ensemble d'articles, son but est d'allouer ces articles de manière à maximiser son revenu.

L'organisation de ce mémoire est la suivante : la première partie dresse un rappel des formulations existantes sur le problème d'enchères combinatoires auxquels nous apportons notre formulation qui est la formulation multi-objectif du problème d'enchères combinatoires. Puis nous citons les principaux travaux effectués sur le modèle d'enchères combinatoires (mono-objectif).

Dans la deuxième partie de ce mémoire, nous proposons une méthode pour résoudre un problème d'enchères combinatoires bi-objectifs qui s'agit d'une méthode hybride, ceci est présenté dans le dernier chapitre. Mais pour ce faire, nous allons commencer par exposer au deuxième chapitre quelques concepts de base de la théorie multicritère, et présenter dans le troisième chapitre un petit panorama des méta-heuristiques existante.

1 Les Enchères Combinatoires

Contents

1.1 Introduction

La théorie des enchères a beaucoup contribué au succès de l'histoire des sciences économiques, sur ses deux plans théorique et pratique : d'un point de vu pratique, elle est importante car les plus importants marchés du monde sont des marchés d'enchères. Elle est aussi théoriquement importante, car les résultats de la théorie des enchères ont mené à de grandes découvertes en sciences de l'économie.

Les enchères ne sont pas une nouvelle idée, plusieurs civilisations ont utilisé le mécanisme d'enchère pour résoudre quelques problèmes : l'enchère des épouses des babyloniens [7], les grecs enchérissaient sur les concessions de mines [27], les romains faisaient des enchères sur les butins de guerre [27]. . . . Dans le monde moderne, les enchères représentent le moteur principal des transactions économiques.

Le premier à avoir analysé les enchères à partir de la théorie des jeux est *Vickery* en 1961 [33]. En 1966, il reçoit le prix Nobel en économie qui est dû à ses travaux en théorie des enchères.

Dans ce chapitre nous allons tout d'abord définir les enchères et citer quelques types d'enchères existantes en économie, puis nous nous intéresserons aux enchères combinatoires. Dans ce chapitre nous développons une formulation mathématique pour le problème d'enchères combinatoires et nous proposons un exemple bien détaillé pour la formulation.

1.2 Problème d'optimisation combinatoire

Un problème d'optimisation combinatoire est défini par un ensemble fini de solutions dites réalisables (formant un espace de décision Ω) et une fonction objectif f associant à chaque solution de l'espace de recherche un coût. Cet ensemble de valeurs constitue l'espace objectif (Y, donc $f : \Omega \rightarrow Y$). Une solution réalisable x peut être donc décrite par un vecteur de décision $(x_1, x_2, \ldots, x_n)$ à laquelle une valeur objectif est associée $f(x)$. Résoudre un problème d'optimisation combinatoire revient à trouver la ou les solutions réalisables de coût maximal (minimal) de Ω.

Définition 1 $x^* \in \Omega$ est une solution optimale si et seulement si :

$$\forall x \in \Omega \; f(x) \geq f(x^*)$$

La difficulté majeure en optimisation combinatoire est due au phénomène d'explosion combinatoire (augmenter la taille de la donnée engendre généralement une augmentation exponentielle de la taille de l'espace de recherche).

1.3 Complexité algorithmique

Un grand nombre de problèmes d'optimisation combinatoire sont NP-difficiles, ce qui signifie qu'il est fort vraisemblable qu'il n'existe pas d'algorithme permettant de les résoudre qui soit efficace, autrement dit qui soit capable de faire significativement mieux, dans *le pire des cas*, que l'examen exhaustif du nombre exponentiel de solutions.

La complexité d'un algorithme est une fonction de la taille du problème (par exemple le nombre de villes dans le cas du problème du voyageur de commerce) qui fournit, à un facteur multiplicatif près, une borne supérieure sur le temps d'exécution de l'algorithme. Dans la mesure où cette borne est indépendante de l'instance, on parle de complexité au pire. On dit qu'un algorithme est *polynomial* ou tout simplement *efficace*, si cette fonction est un polynôme.

Il existe de nombreux problèmes d'optimisation combinatoire qui peuvent être résolus efficacement, on dit aussi en *temps polynomial*. Par exemple, le problème du plus court chemin dans un graphe, le problème du flot de plus petit coût sur un réseau de transport et les problèmes de couplages sont tous des problèmes polynomiaux. Néanmoins, il existe aussi de nombreux problèmes que l'on ne sait pas, aujourd'hui, résoudre efficacement, il s'agit des problèmes NP-difficiles. L'ensemble NP est l'ensemble des problèmes de décisions, c'est-à-dire des questions dont la réponse est soit oui soit non, tels que, pour chacune des instances dont la réponse est oui, il existe un certificat qui permet de montrer en temps polynomial que la réponse est bien oui.

L'ensemble des problèmes de décision que l'on peut résoudre en temps polynomial est noté P et il est généralement conjecturé que $P \neq NP$, autrement dit que certains problèmes de NP ne peuvent pas être résolus en temps polynomial.

Il existe dans NP des problèmes qui peuvent être *réduit polynomialement* à tous les autres problèmes de NP, il s'agit des problèmes NP-complets. L'existence de tels problèmes est riche de conséquences : les problèmes NP-complets sont les plus difficiles de NP et s'il existe un algorithme polynomial qui permet de résoudre l'un des problèmes NP-complets alors cet algorithme peut être utilisé pour résoudre tous les problèmes de NP (NP-complets ou non) en temps polynomial.

1.4 C'est quoi une enchère ?

1.4.1 Définitions

Dans le cas le plus élémentaire et le plus banal, une enchère est une procédure permettant au propriétaire d'un bien unique et indivisible, qui désire le vendre,

de sélectionner l'acquéreur parmi plusieurs candidats (ou, symétriquement, à un acheteur désirant acquérir un objet unique et indivisible de sélectionner le fournisseur parmi plusieurs concurrents). L'objectif du vendeur est ici sensé être d'obtenir le prix de cession le plus élevé possible (il peut se présenter des cas où les objectifs du vendeur sont plus complexes, par exemple savoir entre quelles mains l'objet vendu va tomber, quel usage va en être fait, d'où proviennent les ressources financières de celui qui achète, etc.).

Les enchères se présentent en effet comme un instrument susceptible de renforcer l'efficacité et d'améliorer la transparence en matière de gestion publique. Elles sont utilisées, ou pourraient l'être, dans des domaines aussi variés que l'attribution des Obligations Assimilées du Trésor (OAT), de fourniture d'électricité, voire de sillons ferroviaires congestionnés. Mais elles interviennent aussi pour l'acquisition de produits ou équipements divers par les administrations, ainsi que pour l'attribution de contrats de travaux publics ou de concessions de services publics.

Les formes qu'elles peuvent revêtir sont multiples, allant de l'enchère ascendante la plus connue, qui est celle des salles de vente, à des formes beaucoup plus sophistiquées d'enchères simultanées, en passant par des formes classiques comme l'appel d'offres avec remise de soumissions sous plis scellés. Plus généralement,

> *est une enchère toute procédure qui établit une liaison entre le prix sur lequel s'engage chaque candidat et les prestations qu'il doit réaliser, dont la quantité et la qualité sont clairement spécifiées et doivent être vérifiables*.

Ces spécifications sont le plus fréquemment fournies préalablement aux enchères par le maître d'ouvrage ; le critère d'attribution est alors purement et simplement le prix. Mais dans certains cas, chaque candidat peut être invité à définir lui-même ses prestations en même temps que son offre de prix ; c'est encore une enchère, si l'attribution se fait selon un score qualité-prix chiffré.

1.4.2 Quelques types d'enchères existant en économie

Une forme bien connue est celle de *l'enchère ascendante* (ou enchère *anglaise*), où le prix proposé est augmenté par étapes. Dans cette procédure, tout candidat se désiste au moment où le prix proposé dépasse l'offre maximale qu'il est prêt à faire pour acquérir le bien. Le processus d'élimination s'arrête lorsqu'il

ne reste plus en lice qu'un seul candidat : celui dont l'offre est assurément supérieure à toutes les autres, même si son montant exact, dans cette procédure, reste inconnu. Le bien est attribué à ce candidat " le plus offrant", mais à un prix de cession égal à l'offre la plus élevée parmi celles des candidats éliminés, appelé "deuxième prix".

L'enchère descendante (ou enchère *hollandaise*) existe aussi : le prix proposé, au départ supérieur à l'offre maximale de tous les candidats, est abaissé par étapes, jusqu'à ce qu'un candidat se déclare preneur. Le bien est alors attribué à ce candidat " le plus offrant ", mais à un prix de cession égal à son offre, appelée " premier prix " (les offres des autres candidats restant, dans cette procédure sont inconnues).

Les soumissions sous plis scellés constituent *des enchères à un seul tour*. Les offres de tous les candidats sont cette fois connues pour le vendeur. L'offre est toujours attribuée au candidat le plus offrant. Dans le cas le plus usuel, il doit payer ce bien au premier prix, égal au montant de son offre ; on peut montrer que cette procédure est équivalente à une enchère hollandaise. Mais le règlement de l'appel d'offres peut aussi prévoir que le bien sera payé au deuxième prix, égal au montant du deuxième plus offrant ; cette procédure n'est toutefois pas tout à fait équivalente à l'enchère anglaise.

Pour l'enchère *Vickery* (deuxième-prix, offre-cachée) chaque participant soumet une offre sans savoir les offres des autres, dans un seul tour. Jusqu'à ce moment le protocole est le même que celui de l'enchère premier-prix offre-cachée. La différence est que le participant qui a fait l'offre la plus grande gagne mais il doit payer le prix de la deuxième plus grande offre. La stratégie dominante d'un participant dans ce cas est de soumettre une offre avec sa valeur privée de l'objet. À cause de cette stratégie, l'enchère Vickery est un protocole préféré pour les agents logiciels. Pourtant, elle est moins répandue dans les enchères entre humains à cause du fait que l'initiateur peut mentir sur le deuxième prix le plus élevé et faire payer le gagnant plus que ce prix. Dans les enchères électroniques on peut imposer une signature digitale à toutes les offres des participants et, de cette manière, avoir une possibilité de vérifier le prix à payer.

Les mérites respectifs de ces diverses formes d'enchères dépendent beaucoup des caractéristiques des candidats.

1.4.3 Les enchères combinatoires

Les enchères combinatoires sont des mécanismes où les agents ont la possibilité de lancer des offres non plus simplement sur des objets isolés, mais sur des ensembles d'objets provenant d'une base finie. Elles permettent une expression plus raffinée des préférences des agents et rendent possible une allocation plus efficace des biens.

En enchères combinatoires, le vendeur est soumis à un ensemble de coûts donné pour un ensemble d'articles. Son but est d'allouer ces articles de manière à maximiser son revenu, on dit aussi Problème de Détermination du Gagnant PDG (en anglais Winner Determination problem WDP), fonctionne en choisissant un sous ensemble d'enchères qui maximise le revenu du vendeur.

1.5 Enchères combinatoires

1.5.1 Situations réelles modélisées par les enchères combinatoires

Les enchères combinatoires ne sont pas simplement un casse-tête algorithmique ; il existe de nombreuses applications potentielles, comme par exemple :

- Les allocations de plages de cours à des étudiants (contraintes typiques : les cours ne se chevauchent pas, le nombre d'U.V (unités) est correct, un T.D. (travaux dirigés) pour chaque module, etc...).

- Les allocations de plages de décollage et d'atterrissage (contraintes typiques : avoir un atterrissage pour tout décollage, etc...).

- Les allocations de "travel packages " (chambres d'hôtel, billets d'avion, locations de véhicules... ; contraintes typiques : distance hôtel/aéroport, temps de location du véhicule, ...).

- Aux Etats-Unis, l'attribution d'ondes radios aux compagnies de téléphonie mobile par la Commission Fédérale des Communications (FCC).

Si les applications potentielles des enchères combinatoires sont nombreuses, les raisons d'y recourir le sont également : l'intérêt de ces enchères, c'est de pouvoir (partiellement au moins) capturer une structure de préférence complexe des

agents. En effet,

- Un agent peut être prêt à payer plus, pour deux objets, s'il les obtient tous les deux, que la somme de ce qu'il serait prêt à payer pour chacun pris isolément ; on dit qu'il y a complémentarité entre ces biens.

- Inversement, un agent peut être disposé à payer moins, pour deux objets obtenus conjointement, que la somme de ce qu'il serait prêt à payer pour chacun ; on dit alors qu'il y a substituabilité entre ces biens.

1.5.2 Les modèles existants

Problème de détermination du gagnant en enchères combinatoires

Les enchères combinatoires sont des mécanismes où les agents ont la possibilité de lancer des offres non plus simplement sur des objets isolés, mais sur des ensembles d'objets provenant d'une base finie ; elles permettent une expression plus raffinée des préférences des agents et rendent possible une allocation plus efficace des biens.

En enchères combinatoires, le vendeur est soumis à un ensemble de coûts donné pour un ensemble d'articles, son but est d'allouer ces articles de manière à maximiser son revenu. Le problème de determination du gagnant (PDG, en anglais Winner Determination problem WDP) fonctionne en choisissant un sous ensemble d'enchères qui maximisent le revenu du vendeur.

Définition 2 *Le vendeur possède un ensemble d'articles* $M = \{1, 2, \ldots, m\}$ *à vendre. Les enchérisseurs (ou les soumissionnaires) soumettent un ensemble d'enchères* $B = \{b_1, b_2, \ldots, b_n\}$ *; une enchère est formée d'un couple* $b_j = (S_j, P_j)$ *où* $S_j \subseteq M$ *l'ensemble des articles choisis et* $P_j \geq 0$ *le prix proposé pour cet ensemble d'articles.*

Le problème de determination du gagnant en enchères combinatoires binaires *est l'énumération des enchères gagnantes ou perdantes qui maximisent le revenu du vendeur sous contraintes qu'un article est alloué à au plus un enchérisseur.*

Le modèle mathématiques

$$\begin{cases} \max Z = \sum_{j=1}^{n} x_{b_j} P_j \\ \qquad \sum_{\{j \in \{1,\ 2,\ ...,\ n\} / i \in S_j\}} x_{b_j} \leq 1, \qquad S_j \subseteq M, \quad i = 1,\ 2,\ \ldots,\ m. \\ \\ \qquad x_{b_j} \in \{0,\ 1\}, \qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Ici les x_{b_j} sont définis par :

$$x_{b_j} = \begin{cases} 1, & \text{si l'enchère } b_j \text{ est gagnante,} \\ 0, & \text{sinon.} \end{cases}$$

Formulation du problème du PDG avec contraintes XOR

$$\begin{cases} \max Z = \sum_{j}^{n} P_j x_{b_j} \\ \\ \qquad \sum_{\{j \in \{1,\ 2,\ ...,\ n\} / i \in S_j\}} x_{b_j} \leq 1, \qquad S_j \subseteq M, \quad i = 1,\ 2,\ \ldots,\ m. \\ \\ \qquad \sum_{\{j \in \{1,\ 2,\ ...,\ n\} / S_j \in \gamma_i\}} x_{b_j} \leq 1, \qquad S_j \subseteq M, \ \ \gamma_i \subseteq \Gamma \\ \\ \qquad x_{b_j} \in \{0,\ 1\}, \qquad\qquad\qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Où $\Gamma = \{\gamma_1,\ \gamma_2, \ldots,\ \gamma_m\}$, les sous ensembles d'articles qui ne doivent pas être allouer ensemble.

Formulation du problème du PDG Multi-unités

Dans certaines enchères, il existe de multiples unités du même article à vendre. On peut comprimer les offres et accélérer la détermination du gagnant en ne traitant pas chaque unité comme un article séparé. Puisque les enchérisseurs (soumissionnaires) ne s'inquiètent pas des unités de chaque article qu'ils obtiennent, mais plutôt du nombre d'unités.

On définit alors une enchère par $b_j = ((\lambda_j^1,\ \lambda_j^2,\ \ldots, \lambda_j^m), p_j)$, où $\lambda_j^k \geq 0$ est le nombre d'unités demandées pour l'article $k = 1, ..., m$ et p_j le prix. On définit aussi u_i le nombre d'unités de l'article i.

$$\begin{cases} \max Z = \sum_{j}^{n} P_j x_{b_j} \\ \\ \sum_{j=1}^{n} \lambda_j^i x_{b_j} \leq u_i, \qquad i = 1,\ 2,\ 3, \ldots, m \\ \\ x_{b_j} \in \{0,\ 1\}, \qquad\qquad\qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Formulation du problème d'échanges combinatoires

Dans ce cas d'enchères combinatoires, les acheteurs et les vendeurs peuvent soumettre des offres combinatoires. Les enchères sont comme dans le cas de Multi-unité-PDG, sauf que ici les λ_j^i et p_j peuvent avoir des valeurs negatives (représentation de la vente au lieu de l'achat). Et donc le problème est représenté par :

$$\begin{cases} \max Z = \sum_{j}^{n} P_j x_{b_j} \\ \\ \sum_{j=1}^{n} \lambda_j^i x_{b_j} \leq 0, \qquad i = 1,\ 2,\ 3, \ldots, m \\ \\ x_{b_j} \in \{0,\ 1\}, \qquad\qquad\qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Prix de réserve

Dans certaines enchères le vendeur possède un prix de reserve r_i pour l'article i, qu'il faut obtenir, en dessous duquel il ne vend pas. Ceci n'est pas fait par une contrainte, mais juste en changeant la fonction objectif.

$$\max \sum_{j}^{n} (P_j - \sum_{i \in S_j} r_i) x_{b_j}$$

1.6 Formulation multi-objectif pour le problème de détermination du gagnant

Considérons un problème d'enchères où il peut se présenter plusieurs vendeurs qui proposent chacun une gamme de divers articles à vendre. Nous proposons dans cette section une formulation multi-objectif pour le problème de determination du gagnant.

• Soit K le nombre de vendeurs présent pour l'enchère (on suppose que les vendeurs sont indexés par $q = 1,\ 2,\ \ldots,\ K$).

• Pour chaque vendeur q, soit un ensemble $M_q \subseteq M$, $M_q = \{1,\ 2,\ldots,\ m_q\}$, l'ensemble des articles qu'il propose et $U_q = \{u_{qi}\}_{i \in M_q}$, les nombres d'unités disponibles pour chacun des articles proposés.

• On suppose qu'il y a n enchérisseurs notés : $j = 1,\ 2,\ \ldots,\ n$.

On représente une enchère j par le vecteur :

$$b_j = (\{\lambda_j^{1i}\}_{i \in M_1}, \{\lambda_j^{2i}\}_{i \in M_2}, \ldots, \{\lambda_j^{ki}\}_{i \in M_k}, P_j^1, P_j^2, \ldots, P_j^k)$$

où λ_j^{qi} désigne le nombre d'unités de l'article i demandés par l'enchérisseur j au vendeur q et P_j^q le prix ainsi proposé pour cet ensemble d'articles.

On note x_{b_j} la variable de l'enchère. Ainsi, si $x_{b_j} = 1$ alors elle est gagnante, sinon ($x_{b_j} = 0$) alors elle est perdante.

$$x_{b_j} = \begin{cases} 1, & \text{si l'enchère } b_j \text{ est gagnante,} \\ 0, & \text{sinon.} \end{cases}$$

Comme il y a k vendeurs, la fonction objectif s'écrit sous la forme de plusieurs fonctions à maximiser (une pour chaque vendeur) :

– Pour le vendeur 1 on a : $\max Z^1 = \sum_{j=1}^{n} P_j^1 x_{b_j}$.

– Pour le vendeur 2 on a : $\max Z^2 = \sum_{j=1}^{n} P_j^2 x_{b_j}$.

– ...

– De manière générale, le vendeur $\forall\ k\ \in \{1,\ 2,\ \ldots,\ K\} : \max Z^k = \sum_{j=1}^{n} P_j^k x_{b_j}$.

Et donc ceci revient à maximiser :

$$\max Z^q = \sum_{j=1}^{n} P_j^q x_{b_j}, \quad q = 1,\ 2,\ \ldots,\ k.$$

Sous les contraintes :

$$\begin{cases} \sum_{j=1}^{n} \lambda_j^{1i} x_{b_j} \leq u_{1i}, & i = 1,\ 2,\ldots,\ m_1 \\ \vdots & \\ \sum_{j=1}^{n} \lambda_j^{ki} x_{b_j} \leq u_{ki}, & i = 1,\ 2,\ldots,\ m_K \\ \\ x_{b_j} \in \{0,\ 1\}, \qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Ces contraintes peuvent s'écrire sous la forme :

$$\begin{cases} \sum_{j=1}^{n} \lambda_j^{qi} x_{b_j} \leq u_{qi}, \qquad q = 1,\ 2,\ldots,\ K, \quad i = 1,\ 2,\ldots,\ m_q \\ \\ x_{b_j} \in \{0,\ 1\}, \qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

En résumé, le problème s'écrit :

$$\begin{cases} \max Z = \sum_{j=1}^{n} x_{b_j} P_j^q, \qquad q = 1,\ 2,\ \ldots,\ K, \\ \\ \qquad \sum_{j=1}^{n} \lambda_j^{qi} x_{b_j} \leq u_{qi}, \qquad q = 1,\ 2,\ldots,\ k, \quad i = 1,\ 2,\ldots,\ m_q \\ \\ \qquad x_{b_j} \in \{0,\ 1\}, \qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Remarque 1
A partir du modèle précédent, on peut retrouver le modèle d'enchère combinatoire où la vente est réalisée par un seul vendeur et un ensemble d'articles, le cas d'enchères où plusieurs unités d'articles sont mises en vente etc.

Exemple :

Nous proposons dans ce paragraphe un exemple réel qui illustre un problème de détermination du gagnant dans le cas multi-objectif.

Soient les articles i, $i \in M = \{1,\ 2,\ 3,\ 4,\ 5,\ 6\}$ où M l'ensemble des articles. On suppose qu'il y a 10 exemplaires dans chaque article.

Dans cette enchère, on compte 05 vendeurs aux quels on attribut l'indice q par la suite, $q \in \{1,\ 2,\ 3,\ 4,\ 5\}$, chaque vendeur propose des articles en vente, soit M_q l'ensemble des articles proposés par chaque vendeur.
On a :

$$\begin{cases} M_1 = \{1,\ 2\}, \\ M_2 = \{3\}, \\ M_3 = \{4\}, \\ M_4 = \{5\}, \\ M_5 = \{6\}, \\ M = \bigcup_{q=1}^{6} M_q. \end{cases}$$

On suppose qu'il y a $n = 4$ enchérisseurs $b_1,\ b_2,\ b_3$ et b_4, chaque enchérisseur propose un ensemble d'articles à enchérir. Le tableau suivant résume toutes les enchères soumises :

Enchères	Articles demandés	Nombre d'articles demandés	Prix proposé
b1	(1, 2, 6)	(2, 4, 5)	(50, 10, 20)
b2	(1, 2, 3)	(2, 4, 10)	(40, 10, 10)
b3	(4, 5, 6)	(3, 7, 6)	(100, 30, 60)
b4	(1, 4, 5)	(8, 2, 1)	(100, 40, 5)

En modelant les données de ce tableau, on obtient le modèle mathématique suivant :

• Les variables de décision de ce problème sont : $x_{b_j} \in \{0,\ 1\}$, $j \in \{1,\ 2,\ 3,\ 4\}$.

• Les revenus des vendeurs :

– Pour le vendeur 1 on a : $\max \sum_{j=1}^{n} P_j^1 x_{b_j} \Rightarrow \max f_1(x) = 60x_{b_1} + 50x_{b_2} + 100x_{b_4}$.

– Pour le vendeur 2 on a : $\max \sum_{j=1}^{n} P_j^2 x_{b_j} \Rightarrow \max f_2(x) = 10x_{b_2}$.

– Pour le vendeur 3 on a : $\max \sum_{j=1}^{n} P_j^2 x_{b_j} \Rightarrow \max f_3(x) = 100x_{b_3} + 40x_{b_2}$.

– Pour le vendeur 4 on a : $\max \sum_{j=1}^{n} P_j^2 x_{b_j} \Rightarrow \max f_4(x) = 30x_{b_3} + 5x_{b_4}$.

– Pour le vendeur 5 on a : $\max \sum_{j=1}^{n} P_j^2 x_{b_j} \Rightarrow \max f_5(x) = 20x_{b_1} + 60x_{b_3}$.

• Pour que les demandes ne chevauchent pas et aussi pour rester dans ce qui est disponible comme articles, on doit vérifier quelques contraintes :

$$\begin{cases} 2x_{b_1} + 2x_{b_2} + 8x_{b_4} \leq 10, \\ 4x_{b_1} + 4x_{b_2} \leq 10, \\ 10x_{b_2} \leq 10, \\ 3x_{b_3} + 2x_{b_4} \leq 10, \\ 7x_{b_3} + x_{b_4} \leq 10, \\ 5x_{b_1} + 6x_{b_3} \leq 10, \\ x_{b_j} \in \{0,\ 1\}, \quad \forall j \in \{1,\ 2,\ 3,\ 4\} \end{cases}$$

Et le modèle s'écrit donc :

$$\begin{cases} \max f_1(x) = 60x_{b_1} + 50x_{b_2} + 100x_{b_4} \\ \max f_2(x) = 10x_{b_2} \\ \max f_3(x) = 100x_{b_3} + 40x_{b_2} \\ \max f_4(x) = 30x_{b_3} + 5x_{b_4} \\ \max f_5(x) = 20x_{b_1} + 60x_{b_3} \\ S.c \\ 2x_{b_1} + 2x_{b_2} + 8x_{b_4} \leq 10, \\ 4x_{b_1} + 4x_{b_2} \leq 10, \\ 10x_{b_2} \leq 10, \\ 3x_{b_3} + 2x_{b_4} \leq 10, \\ 7x_{b_3} + x_{b_4} \leq 10, \\ 5x_{b_1} + 6x_{b_3} \leq 10, \\ x_{b_j} \in \{0,\ 1\}, \quad \forall j \in \{1,\ 2,\ 3,\ 4\} \end{cases}$$

1.6.1 Comment retrouver les modèles existants

Formulation du problème de détermination du gagnant -un seul vendeur-

A partir du modèle précédent, on peut retrouver le modèle d'enchères combinatoires où la vente est réalisée par un seul vendeur et un ensemble d'articles en vente.

On pose $K = 1$ (le nombre de vendeurs) et m le cardinal de l'ensemble des articles M, avec un seul exemplaire de chacun $u = 1$ (nombre d'unité).
On suppose qu'il y a n enchérisseurs, alors une enchère est représentée par :

$$b_j = ((\lambda_j^{11},\ \lambda_j^{12},\ \ldots,\ \lambda_j^{1m}),\ P_j^1)$$

Les $\lambda_j^{1q} = 0, \forall j > 1,\ \ q > 1$ car il y a un seul exemplaire pour tous les articles et aussi un seul vendeur, donc l'enchère s'écrit : $b_j = (S_j, P_j)$. Où $S_j \in M$ l'ensemble des articles choisis par l'enchérisseur j.

La fonction objectif à optimiser est : $\max Z = \sum_{j=1}^{n} x_{b_j} P_j$.

Et sous les contraintes :

$$\begin{cases} \sum\limits_{\{j\in\{1,2,\ldots,n\}\ |\ i\in S_j\}} x_{b_j} \leq 1, \qquad i = 1,\ 2,\ \ldots,\ m. \\ \\ x_{b_j} \in \{0,\ 1\}, \qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Et donc le problème s'écrit :

$$\begin{cases} \max Z = \sum\limits_{j=1}^{n} x_{b_j} P_j, \\ \\ \qquad \sum\limits_{\{j\in\{1,2,\ldots,n\}\ |\ i\in S_j\}} x_{b_j} \leq 1, \qquad i = 1,\ 2,\ \ldots,\ m. \\ \\ \qquad x_{b_j} \in \{0,\ 1\}, \qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Formulation du problème de détermination du gagnant -Multi-unités-

De la même manière on peut retrouver le problème d'enchères combinatoires multi-unités à partir du modèle précédent.

Ainsi, on a $K = 1$ (un seul vendeur). Dans ce cas on peut avoir plusieurs unités pour les articles misent en vente donc $U = \{u_1,\ u_2, \ldots,\ u_m\}$, où m le nombre d'articles misent en vente et u_i le nombre d'exemplaire pour chaque articles.

Alors une enchère est représentée par : $b_j = ((\lambda_j^1, \lambda_j^2, \ldots, \lambda_j^m), P_j)$, où $\lambda_j^i \geq 0$ est le nombre d'unités de l'article i demandé par l'enchérisseur j.

La fonction objectif à optimiser est : $\max \sum_{j=1}^{n} x_{b_j} P_j$.

Sous les contraintes :

$$\begin{cases} \sum_{j=1}^{n} \lambda_j^i x_{b_j} \leq 1, & i = 1, 2, \ldots, m. \\ \\ x_{b_j} \in \{0, 1\}, & j = 1, 2, \ldots, n \end{cases}$$

La formulation du problème est donc :

$$\begin{cases} \max Z = \sum_{j=1}^{n} x_{b_j} P_j, \\ \\ \quad \sum_{j=1}^{n} \lambda_j^i x_{b_j} \leq 1, & i = 1, 2, \ldots, m. \\ \\ \quad x_{b_j} \in \{0, 1\}, & j = 1, 2, \ldots, n \end{cases}$$

1.7 Relation entre le problème du PDG et d'autres problèmes d'optimisation

PDG et problème du sac à dos multi-dimensionnels

Le problème du sac à dos (Knapsack problem) est un problème d'optimisation largement connu. Ce problème est NP-Complet (*Karp*, 1972) [33]. Dans sa forme standard le problème sac à dos est formulé par :

$$\begin{cases} \max Z = \sum_{j=1}^{n} x_j P_j \\ \quad \sum_{j}^{n} r_j x_j \leq c_i, & i = 1, 2, \ldots, m. \\ \\ \quad x_j \in \{0, 1\}, & j = 1, 2, \ldots, n \\ \quad P_j > 0, \, c_i, \, r_j \in \mathbb{N}. \end{cases}$$

Le problème du sac à dos multi-dimensionnels (MDKP) est connu sous le nom problème du sac à dos à d-dimensions, problème du sac à dos multi-contraintes ou aussi problème du sac à dos multiple, sa formulation en programmation en nombres entiers est la même que celle du problème de determination du gagnant (PDG) multi-unités (*Pfeiffer*, 2006) [33]. Cette relation entre ces deux problèmes (MDKP et PDG multi-unités) est initialement discuter par *Holte* en 2001 [33].

PDG et problème de remplissage (Weighted set packing)

Le problème de remplissage (Weighted set packing problem) est définit comme suit : Soit un un ensemble M et une collection V de sous-ensembles de poids non négative, donc le problème est de trouver une collection de poids max avec des sous-ensembles disjoints. Sa formulation en programmation en nombres entiers est :

$$\begin{cases} \max Z = \sum_{j \in V} x_j C_j \\ \quad \sum_{j \in V} a_{ij} x_j \leq 1, \qquad i = 1,\ 2,\ \ldots,\ m. \\ \\ \quad x_j \in \{0,\ 1\}, \qquad j = 1,\ 2,\ \ldots,\ n \end{cases}$$

Le problème de determination du gagnant PDG est une instance du problème WSP (weighted set packing) [9, 30].

Remarque 2
Ces deux equivalences données ici pour le problème d'enchères combinatoires ont été prouver pour le cas d'enchères combinatoires mono-objectif. Dans le cas des enchères combinatoires multi-objectif, on peut remarquer que lorsque le nombre d'articles demandés par un enchérisseur est souvent le même pour tout les articles, là le problème peut être aussi dit équivalent au problème du sac à dos multi-objectif. Nous utiliserons cette propriété au chapitre 4 (Résolution du problème d'enchères Combinatoires Bi-objectifs par une Méthode Hybride).

1.8 Complexité du problème PDG

Le problème de determination du gagnant est facile dans le cas d'enchères non-combinatoires, car il se fait en prenant directement la plus grande offre pour chaque article séparément. Ceci prend en temps d'exécution $O(nm)$, où n est le nombre d'enchères et m est le nombre d'articles.

Lorsqu'il s'agit d'enchères combinatoires, le problème de determination du gagnant (PDG) est dur parce qu'on devrait vérifier, pour chaque sous-ensemble d'une enchère, si le sous-ensemble est faisable ou non (risque d'avoir des chevauchements dans les sous-ensembles d'articles) et combien chaque sous-ensemble fournit comme revenu pour le vendeur.
Un sous-ensemble faisable qui rapporte le plus haut revenu est une solution optimale.

Malheureusement, il y a 2^k sous-ensembles d'offres, k étant le nombre d'offres, ainsi l'énumération est infaisable (excepté lorsqu'il y a moins de 30 offres) [30]. Ce problème est NP-complet et inapproximable (non-approximable) [30]. Ce résultat peut être renforcé par le fait que ce problème est équivalent au problème de remplissage d'ensemble (weighted set packing), qui est connu NP-Complet.

1.9 Travaux antérieurs

Le premier algorithme proposé pour retrouver une allocation optimale pour le problème de determination du gagnant (PDG un seul vendeur, un seul article) a été publié par *Rassenti et al* (1982)[33]. Ils ont considéré le problème d'enchères dans le contexte de vente des temps d'atterrissage dans les aéroports afin de permettre à des compagnies aériennes d'offrir simultanément des temps de décollage et d'atterrissage.

Après, plusieurs algorithmes ont été proposés pour résoudre le problème de détermination du gagnant (PDG) avec une seule unité pour chaque article en vente. Parmi ces méthodes nous citons : *Hoos et boutilier* (2000), *Sakurai et al* (2000) [33], l'algorithme de Branch-On-Item (BoI) *Sandholm. T et Suri. S* [36], Branch-On-Bids (BoB) *Sandholm. T et Suri. S* [35], *Sandholm. T, Suri. S, Gilpin. A et Levine. D* (CABoB) [36], Combinatorial Auctions Structural search (CASS) [31], un algorithme de Branch and Bound proposé par *Fujishima et al* 1999 [4].

Pour la résolution du problème (PDG) Multi-unités on retrouve : *Leyton-Brown et al* 2000, *Holte* 2001, *Gonen at Lehmann* 2002 [33], *Leyton-Brown, K., Shoham, Y., et Tennenholtz, M.*(CAMUS) [32], une approche de programmation dynamique a été proposée par *Rothkopf et al* 1998 [4], l'approche de programmation linéaire a été proposée par *Nisan* 2000[23] et autres approches existent pour la résolution de ce problème d'enchères combinatoires [4, 33].

1.10 Conclusion

Dans ce premier chapitre, nous avons abordé la définition de l'enchère économique, cité les plus importants types d'enchères : les enchères Anglaise, Hollandaise, Vickery etc. Puis nous avons rappelé aussi la définition d'un problème d'enchères combinatoires connu sous le nom de problème de détermination du gagnant PDG.

En deuxième lieu, nous avons proposé une nouvelle formulation mathématique pour le problème d'enchères combinatoires, qui prend en compte une enchère avec plusieurs vendeurs et différents articles. Cette formulation multi-objectif du problème d'enchères combinatoires représente un schéma plus général et plus complet du problème d'enchères combinatoires, à partir duquel on peut déduire des cas plus élémentaires cités dans la littérature.

Dans la suite de ce mémoire, nous allons construire une méthode hybride pour résoudre ce problème d'enchères combinatoires.

2

Concepts de base de l'optimisation multi-critère

Contents

2.1 Introduction

Résoudre un problème d'optimisation combinatoire consiste à trouver une solution qui optimise une fonction appelée *fonction objectif*. Pendant longtemps, la plupart des travaux d'optimisation étaient dédiés à l'optimisation d'une seule fonction objectif. Or la plupart des problèmes rencontrés dans l'industrie (mécanique, télécommunications, environnement...) sont de nature multi-objectif. Il y a généralement plusieurs objectifs contradictoires à optimiser simultanément (coût, qualité, temps, charge,...etc).

Les travaux prenant en compte tous les objectifs on été développés pour la première fois durant le $19^{\text{ème}}$ siècle dans les travaux en économie de *Edgwarth et Pareto*. Ils ont été utilisés initialement en économie et dans les sciences de management et graduellement dans les sciences de l'ingénieur. Différentes définitions ont été proposées pour l'aide multicritère à la décision. Nous reprenons ici celle de Vincke (1989, [6]) :

> *L'aide multicritère à la décision vise, comme son nom l'indique, à fournir à un décideur des outils lui permettant de progresser dans la résolution du problème de décision ou plusieurs points de vue, souvent contradictoires, doivent être pris en compte.*

L'optimisation multi-objectif ou l'aide à la décision multicritère est un domaine dont l'importance est justifiée par la nature multi-objectif des problèmes d'optimisation à résoudre dans la réalité. Elle se propose de traiter les problèmes qui nécessitent la satisfaction de plusieurs critères en même temps. Ces critères sont parfois conflictuels et parfois complémentaires. Un des aspects caractérisant l'optimisation multi-objectif est qu'elle fournit un ensemble de solutions réalisant le meilleur compromis entre les critères considérés. Contrairement à l'optimisation mono-objectif, la solution d'un problème multi-objectif n'est pas unique, mais un ensemble de solution, qui offrent un compromis entre les différents objectifs optimisés, et le choix d'une de ces solutions est alors réalisé par un décideur.

En optimisation combinatoire multi-objectif, un défi majeur est celui de développer des procédures efficaces pour générer des solutions efficaces. Celles-ci ont la propriété qu'aucune amélioration n'est possible sur un objectif sans sacrifier la performance sur au moins un autre objectif. Le but est celui de trouver l'ensemble efficace (constitué de toutes les solutions efficaces) ou, plus fréquemment, un en-

semble efficace réduit (constitué par une solution efficace pour chaque vecteur critère non dominé).

Ce chapitre a pour objectif de présenter le contexte de l'optimisation multi-objectif et ses principaux fondements.

2.2 Problématique

En optimisation multi-objectif, on dispose de p fonctions de valuation ($p \geq 2$) $f_1, f_2, \ldots, f_p$ à valeurs dans $\mathbb{R}$ (souvent $\mathbb{N}$ en optimisation combinatoire). Chaque alternative ou solution réalisable $x \in A$ est associée à un vecteur à p composantes appelé vecteur des performances.

Un problème d'optimisation multi-objectif peut être formulé mathématiquement par :

$$\begin{cases} \min(f_1(x), f_2(x), \ldots, f_p(x)) \\ x \in A. \end{cases} \tag{2.1}$$

Comme il vient d'être souligné, ces problèmes sont dits *mal posés* par le fait que leur solution optimale de pareto (en terme de coût) n'est habituellement pas unique. Ceci nous mène à trois problématiques distinctes et auxquelles on pourrait s'intéresser particulièrement :

a. La recherche d'un compromis (efficace) spécifique : si l'on veut extraire un ensemble efficace supposé le plus "proche" des préférences du décideur.

b. La résolution complète : ceci revient à énumérer complètement l'ensemble de toutes les solutions efficaces ou son ensemble réduit associé.

c. Parce que ce dernier peut être de cardinal exponentiellement grand, on peut s'intéresser à l'ensemble des solutions efficaces possédant une certaine propriété (problèmes restreints)

2.3 Définitions

2.3.1 Minimum (Maximum)

Considérons le problème mono objectif suivant

$$\begin{cases} \min f(x) \\ x \in X. \end{cases} \tag{2.2}$$

Définition 3 La fonction objectif est la fonction f, appelée aussi *fonction coût* ou *critère d'optimisation*, à valeur numérique scalaire ou vectorielle à base de laquelle des alternatives sont comparées par une relation d'ordre complète ou partielle, selon leurs performances (valeur de la fonction objectif).

En général, le terme objectif est employé lorsque l'on mesure une notion modélisée qui a une nature quantitative (coût, distance,...) ; tandis que le terme critère correspond plus à une notion qualitative (appartenance à une classe, relation entre différents objets,...).

Définition 4 *Minimum global*, un point $x \in X$ est un minimum global du problème (2.2) si et seulement si : $\forall x' \in X, f(x) \leq f(x')$ (ensemble des minimums).

Définition 5 *Minimum local*, un point $x \in X$ est un minimum local du problème (2.2) si et seulement si : $\forall x' \in N(X), f(x) \leq f(x')$ où $N(X)$ définit un voisinage de x.

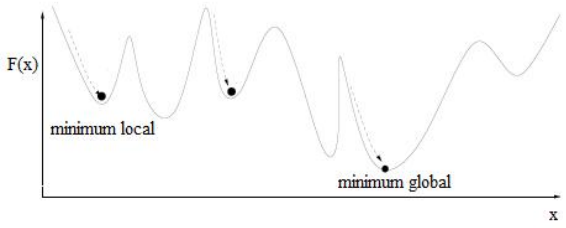

FIG. 2.1 – Minimum local et global.

2.3.2 Espace des décisions - espace des critères

On appelle espace des décisions l'ensemble des alternatives possibles. Il est aussi appelé ensemble des solutions réalisables. Il peut être fini, dénombrable ou infini, discret ou continu. Pour un espace des décisions A, on notera Z_A l'espace des critères associé défini par :

$$Z_A = \{z \in Z \subset \mathbb{R}^p | z_j = f_j(x), x \in A\}, j = 1,\ 2,\ \ldots,\ p. \tag{2.3}$$

Une formulation, équivalente à celle donnée dans le système (2.1), d'un problème d'optimisation multi-critère est la suivante :

$$\begin{cases} \min z = (z_1, z_2, \ldots, z_p) \\ z \in Z_A. \end{cases} \tag{2.4}$$

2.3.3 Non dominance, efficacité

La notion d'optimalité (dite de dominance) la plus généralement admise est celle introduite par *Edgeworth* en 1881, généralisée par *Pareto* en 1896 [2]. Le terme le plus employé pour s'y référer est celui d'*optimum de Pareto*.

Dans l'espace des critères, on définit la relation de dominance comme suit :

Définition 6 Soit $Z \subset \mathbb{R}^p$ et $z, z' \in Z$. On dit que,

a. z *domine faiblement* z' et on note $z \preceq z'$ si et seulement si $z_j \leq z'_j \quad \forall j \in \{1, \ldots, p\}$.

b. z *domine* z' et on note $z \prec z'$ si et seulement si $\begin{cases} z \preceq z' \\ z \neq z' \end{cases}$

c. z *domine strictement* z' et on note $z \ll z'$ si et seulement si $z_j < z'_j \quad \forall j \in \{1, \ldots, p\}$.

Parallèlement, on définit la relation de dominance entre les alternatives comme suit.

Définition 7 Soit $a, b \in A$, z_a, z_b leurs vecteurs critères correspondants dans Z_A. On dit que,

a. *a domine faiblement b* et on note $a \underline{\triangle} b$ si et seulement si $z_a \preceq z_b$.

b. *a domine b* et on note $a \triangle b$ si et seulement si $z_a \prec z_b$.

On peut déduire la relation suivante :

$$a \triangle b \Leftrightarrow (a \underline{\triangle} b) \wedge \neg (b \underline{\triangle} a).$$

Définition 8 $z \in Z_A$ est dit :

a. *non dominé* si et seulement si $\nexists z' \in Z_A$ tel que $z' \prec z$.

b. *faiblement non dominé* si et seulement si $\nexists z' \in Z_A$ tel que $z' \ll z$.

$a \in A$ est dite solution (faiblement) efficace si et seulement si son vecteur critère correspondant $z_a \in Z_A$ est (faiblement) non dominé.

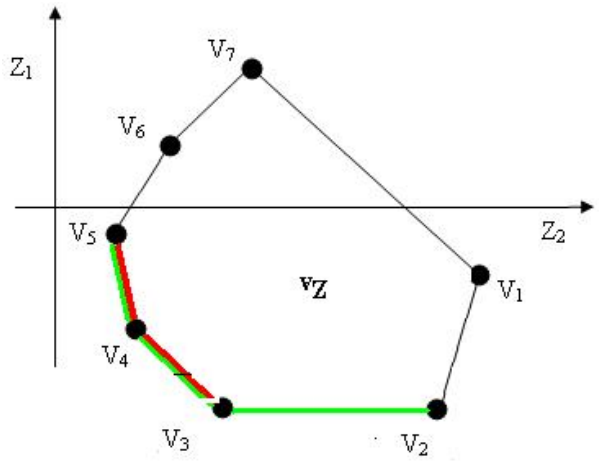

FIG. 2.2 – Exemple d'une frontière de Pareto.

$ND(Z_A) = [z_3, z_4] \cup [z_4, z_5]$ et $FND(Z_A) = ND(Z_A) \cup [z_2, z_3]$.

Ainsi, toute solution efficace (optimale de Pareto) peut être considérée comme optimale de pareto puisque aucune amélioration ne peut être faite sur un objectif sans dégrader la performance sur un autre objectif. Ces solutions forment le *front Pareto*.

Dans le cas d'un problème bi-objectif, les solutions efficaces peuvent être identifiées visuellement dans l'espace objectif.

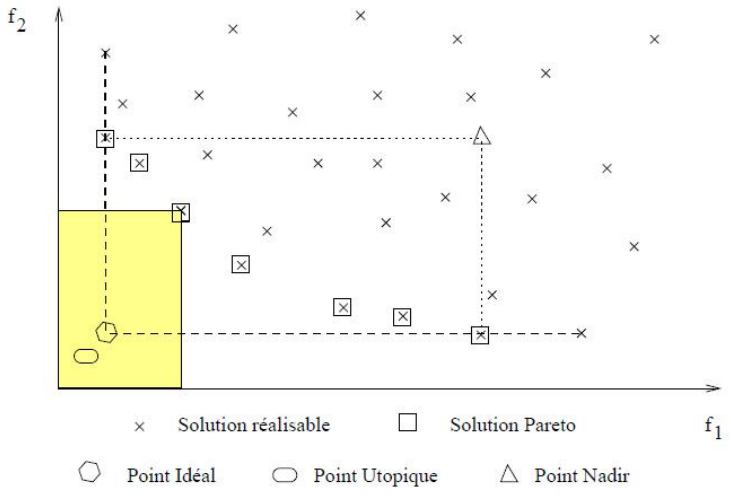

FIG. 2.3 – Cas bi-objectif.

2.3.4 Points particuliers

Notons $E(A)$ l'ensemble des solutions efficaces de A, dont l'image est l'ensemble des vecteurs coût non dominés $ND(Z_A)$.

Point idéal

Définition 9 Le point Idéal z^I est le point qui a comme valeur pour chaque objectif, la valeur optimale de l'objectif considéré.

$$z^I, \forall i \in \{1,2,\ldots,n\} : f_i(z^I) = \min_{x \in A} f_i(x)$$

Ce point ne correspond en général pas à une solution réalisable car si c'était le cas, cela sous-entendrait que les objectifs ne sont pas contradictoires et qu'une solution optimisant un objectif, optimise simultanément tous les autres, ce qui ramènerait le problème à un problème ayant une seule solution Pareto optimale.

Proposition 1

$$\forall i \in \{1,2,\ldots,p\} : f_i(z^I) = \min_{x \in A} f_i(x)$$

Le calcul du point idéal est donc simple et se fait par une optimisation monodimensionnelle.

Point nadir

Définition 10 Le point Nadir est défini par :

$$z^N, \forall i \in \{1,2,\ldots,p\} : f_i(z^N) = \max_{x \in A} f_i(x)$$

Mais on a généralement

$$z^N, \forall i \in \{1,2,\ldots,p\} : f_i(z^N) \neq \max_{x \in A} f_i(x)$$

z^N n'est pas facile à calculer, on a souvent recours à une approximation en utilisant la méthode du *tableau des gains*.

Efficaces supportées et non supportées

L'ensemble Pareto optimal peut être divisé en deux sous-ensembles : l'ensemble des solutions supportées et l'ensemble des solutions non supportées. Les solutions supportées sont les solutions Pareto optimales dont le point correspondant se trouve sur l'enveloppe convexe des solutions réalisables. Les solutions

Pareto n'appartenant pas à l'enveloppe convexe sont dites non-supportées (voir figure suivante).

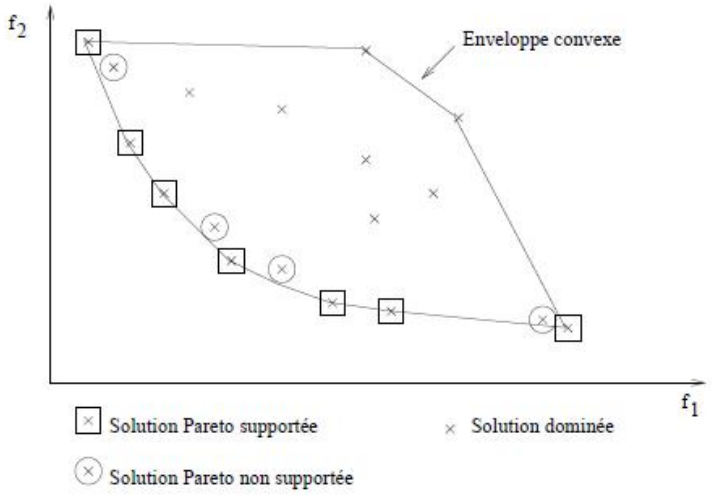

FIG. 2.4 – Efficaces supportés et non supportés.

En fonction du problème étudié et de la connaissance de la structure de sa frontière Pareto, différentes stratégies peuvent être adoptées.

- *La structure du problème est bien connue et sa fonction a été montrée convexe* : toutes les solutions Pareto du problème sont supportées donc une recherche par agrégation linéaire des objectifs est suffisante.

- *La structure du problème est bien connue et sans être convexe, il est montré que les solutions supportées sont bien réparties* : si l'objectif est seulement d'obtenir le front Pareto une agrégation suffit.

- *La structure du problème n'est pas ou est mal connue, ou les solutions supportées sont mal réparties sur le front :* il est nécessaire de rechercher toutes les solutions supportées et non supportées.

2.4 Approches de résolution

L'optimisation multi-objectif ou multicritère est un domaine dont l'importance est justifiée par la nature multi-objectif des problèmes d'optimisation à résoudre

dans la réalité. Elle se propose de traiter les problèmes qui nécessitent la satisfaction de plusieurs critères en même temps. Ces critères sont parfois conflictuels et parfois complémentaires. Un des aspects caractérisant l'optimisation multi-objectif est qu'elle fournit un ensemble de solutions réalisant le meilleur compromis entre les critères considérés.

Il existe plusieurs manières d'approcher un problème d'optimisation combinatoire multi objectif. Ces méthodes sont considérées soit de type exact, approximatif ou interactif. Les méthodes interactives peuvent être considérées comme méthodes exactes ou approximatives. Elles dépendent du décideur.

2.4.1 Méthodes exactes

Les méthodes exactes sont principalement employées lorsque le problème est bien spécifique. Il peut également être celui en raison de l'information supplémentaire fournie par le problème spécifique qu'il est plus facile de produire une méthode exacte.

Il y a très peu de travaux sur les méthodes exactes dans le contexte de la résolution des problèmes d'optimisation multi-objectif, sans doute, à cause de la grande difficulté de ce type de problème. Les références existantes et qui présentent la plupart des méthodes exactes sont *Ulungu* [40] et *Ehrgott et Gandibleux 2000* [15].

Ces méthodes sont basées principalement sur les procédures de *Branch and Bound, Cut ou Price* et la programmation dynamique. Une approche particulière pour l'optimisation multi-objectif est la programmation par but.

Branch and Bound :

Le *Branch and Bound* est une grande classe d'algorithmes et d'où sont dérivées le *Branch and Cut, Branch and Price*. Il utilise la stratégie diviser pour régner, ceci en partitionnant l'espace des solutions en sous espaces pour les optimiser chacun individuellement.

L'idée de base d'un *Branch and Bound* est de construire un arbre où les solutions se forment de manière incrémentale lorsqu'on s'enfonce dans l'arbre. Afin de résoudre plus rapidement le problème, il faut définir une borne supérieure et une autre inférieure. La racine de l'arbre construite par le *Branch and Bound* correspond à une solution partielle vide, les feuilles correspondent à des solutions

réalisables. D'une manière un peu classique, on fixe à chaque niveau de l'arbre une variable de décision. La profondeur de l'arbre nécessaire correspond au nombre de variables de décision à fixer.

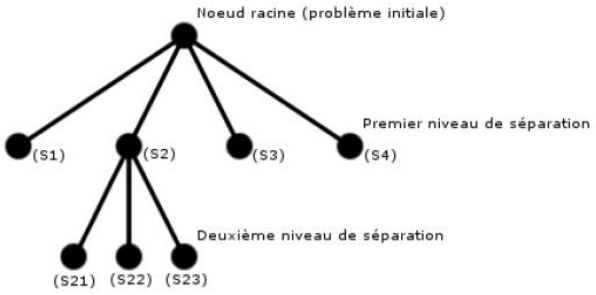

FIG. 2.5 – Arbre généré par par décomposition en sous problème

L'efficacité d'un *Branch and Bound* dépend en grande partie de la borne inférieure et de la stratégie de parcours de l'arbre (*profondeur d'abord, largeur d'abord...*). Une borne inférieure la plus précise possible est nécessaire afin de couper rapidement un maximum de branches de l'arbre, et la stratégie d'exploration doit permettre de découvrir rapidement de bonnes solutions réalisables, afin d'avoir une bonne borne supérieure, toujours dans le but de couper le maximum de branches. L'algorithme du *Branch and Bound* est basé sur la technique de séparation et évaluation.

Principe d'évaluation :
Le principe d'évaluation permet de diminuer l'espace de recherche. L'objectif est d'essayer d'évaluer l'intérêt de l'exploration d'un sous-ensemble de l'arborescence.

Principe de séparation :
Le principe de séparation permet d'établir l'ensemble des règles qui vont régir la séparation d'un ensemble en sous-ensembles. Afin d'assurer l'optimalité de la solution fournie par le *Branch-and Bound*, certaines règles doivent être respectées.

Pedberg et Rinaldi [2] ont amélioré l'idée de *Branch and Bound* basé sur la programmation linéaire en décrivant une méthode utilisant des inégalités renforçant la relaxation par la technique *Branch and Cut*. Depuis beaucoup d'implémentations ont été réalisées en utilisant cette approche.

Les algorithmes de *Branch and Bound* résolvant des programmes linéaires en nombres entiers en générant les variables dynamiquement quand cela est nécessaire sont appelés algorithmes de *Branch and Price*.

Dans le cas des problèmes bi-objectifs, la solution exacte est souvent trouvée par une méthode en deux phases mais toujours implique un certain *Branch and Bound* dans la résolution [38] (Ce point sera développé dans la section suivante).

2.4.2 Les méthodes interactives

Les méthodes interactives incluent le dialogue avec le décideur, car la première étape de calculs utilisée dans ces méthodes fournit une ou plusieurs solutions de compromis, et celles-ci sont présentées au décideur qui réagit en apportant des informations supplémentaires sur ses préférences. Ce processus s'arrête lorsque le décideur se montre satisfait. Ces méthodes réaffirment les racines de la théorie de prise de décision multicritère (MCDM). Elles peuvent être construites en se basant sur une méthode exacte ou approximée ou spéciale, juste qu'elles requièrent l'avis du décideur pour le choix de la solution ou de la région de recherche de solutions.

2.4.3 Les méthodes approximatives

Dés que le nombre d'objectifs ou la taille des problèmes augmentent, les méthodes exactes deviennent inefficaces étant donné la nature NP-difficile des problèmes (mono-objectif) et l'aspect multi-objectif des problèmes. Ainsi, il est nécessaire afin de résoudre des problèmes de grande taille ou avec plusieurs objectifs, de faire appel à des méthodes approximatives. Les méthodes approximatives sont les plus populaires. Elles ne garantissent pas de trouver de manière exacte tout l'ensemble des solutions Pareto , mais une approximation, aussi bonne que possible de cet ensemble.

Afin de rendre les algorithmes plus performants, de nombreux travaux proposent de combiner différentes heuristiques et/ou méthodes exactes. Le but des ces approches est de combiner les avantages des différentes méthodes d'optimisation.

2.5 Les problèmes bi-objectif

En optimisation combinatoire multi-objectif, le cas bi-objectifs est souvent considéré distinctement des cas avec plusieurs objectifs. Malgré le fait que plus d'objectifs fournissent un plus grand défi pour le calcul, une compréhension complète du cas bi-objectifs est essentielle pour faire des pas dans le cas multi-objectif.

Le cas des problèmes bi-objectif est déduit directement du modèle général des problèmes multi-objectif :

$$\begin{cases} \min \ F(x) = (f_1(x),\ f_2(x)) \\ x \in D \end{cases}$$

x est le vecteur représentant les variables de décision, D représente l'ensemble des solutions réalisables et chacune des fonctions $f_i(x)$ est à minimiser (la maximisation se déduit d'une manière triviale).

Une solution z domine la solution z' si, $f_1(z) \leq f_1(z')$ et $f_2(z) \leq f_2(z')$ avec au moins une inégalité stricte. z est dite solution efficace s'il n'existe pas de z' qui domine z.

Méthode des deux phases :
Presque toutes les méthodes proposées pour la résolution du problème bi-objectif sont construites en deux phases : la première phase consiste à chercher les solutions Pareto supportées tandis que la deuxième démarre toujours de ces points trouvés en première phase pour calculer les solutions Pareto non supportées, nous citerons ici quelques unes de ces méthodes.

• La méthode des deux phases a initialement été proposée par *Ulungu et Teghem* pour la résolution d'un problème d'affectation bi-objectif [11] pour chercher toutes les solutions efficaces : la première phase consiste à trouver toutes les solutions supportées du front Pareto, puis la deuxième phase cherche entre ces solutions les solutions Pareto non supportées.

• *Ramos et al*[11] ont présenté une méthode exacte en deux phases pour le problème de l'arbre couvrant minimum, leur papier ne s'est pas limité à la recherche des solutions efficaces mais aussi à énumérer tout les arbres correspondant à ces solutions. Pour la deuxième phase ils ont utilisé un Branch and Bound.

• *Visé et al*[41] appliquent la méthode des deux phases pour le problème de sac à dos bi-objectif. Pour ce faire, ils ont étudié le Branch and Bound avec ces deux versions : '*largeur d'abord*' et '*profondeur d'abord*'. Le résultat de leur expérience a montré que le nombre de solution non-supportées est très grand par rapport à celui des solutions supportées.

• *S. Steiner et E. Talbi* (2006) [38] développent une méthode exacte pour les problèmes bi-objectif en utilisant K-best algorithm, et testent la méthode sur plusieurs problèmes bi-objectif : problème de l'arbre couvrant minimum, perfect matching problem et minimum s-t cut problem. Ils ont aussi prolongé leurs travaux à l'étude de cas tri-objectif.

• *X. Delorme, X. Gandibleux et F. Degoutin* [10] ont proposé pour la résolution du problème set-packing bi-objectif deux approches de résolutions, la première est basée sur une heuristique évolutionnaire multi-objectif basée sur SPEA [1], la deuxième est une adaptation au cas bi-objectif d'une méta-heuristique mono-objectif.

2.6 Conclusion

Nous avons présenté dans ce chapitre les principaux concepts de l'optimisation multi-objectif. Nous avons présenté aussi les trois axes principaux de l'approche de résolution des problèmes multi-objectif : Exactes, Interactives et Approchées. Parmi les méthodes exactes nous avons rappelé le principe générale de la méthode de *Branch and Bound* et les deux méthodes qui dérivent de celle-ci.

En dernier lieu, quelques méthodes intéressantes qui ont donné de bons résultats dans le cas des problèmes bi-objectifs sont présentés.

[1] SPEA : Strength Pareto Evolutionary Algorithm

3
Les Méta-Heuristiques

Contents

3.1 Introduction

L'utilisation des méta-heuristiques est apparue comme une issue face à des problèmes combinatoires dont les espaces de recherche explosent en taille. Pour les problèmes NP-difficiles, en admettant que $P \neq NP$, il n'existe pas d'algorithme en temps polynomial et donc une résolution complète entraîne un temps de calcul exponentiel dans le pire des cas.

Le développement récent de ces méthodes témoigne d'une part de l'intérêt que de nombreux chercheurs y portent, mais aussi du potentiel de ces paradigmes de résolution. Les méta-heuristiques sont basées sur plusieurs critère dont nous citerons quelques uns ici :

◦ Les méta-heuristiques sont des stratégies qui permettent de guider la recherche d'une solution optimale.

◦ Le but visé par les méta-heuristiques est d'explorer l'espace de recherche efficacement afin de déterminer des solutions (presque) optimales.

◦ Les techniques qui constituent des algorithmes de type méta-heuristique vont de la simple procédure de recherche locale à des processus d'apprentissage complexes.

◦ Les méta-heuristiques sont en général non-déterministes et ne donnent aucune garantie d'optimalité.

◦ Les méta-heuristiques peuvent faire appel à des heuristiques qui tiennent compte de la spécificité du problème traité, mais ces heuristiques sont contrôlées par une stratégie de niveau supérieur.

◦ Les méta-heuristiques peuvent faire usage de l'expérience accumulée durant la recherche de l'optimum, pour mieux guider la suite du processus de recherche.

Dans ce chapitre nous allons présenter quelques méta-heuristiques définies dans la littérature pour les problèmes combinatoires mono-objectif et aussi multi-objectif.

3.2 Notions de base

Méta-heuristiques

Les Méta-heuristiques sont des techniques puissantes appliquées a un grand nombre de problèmes combinatoires. Les méta-heuristiques réfèrent à la stratégie itérative qui guide et modifie les opérations et les heuristiques en combinant d'une manière intelligente les différents concepts d'exploration et d'exploitation de l'espace de recherche (*Glover et Laguna* 1997 [12], et *Osman et Laporte* 1996 [13]).

La famille des méta-heuristiques contient, mais ne s'y limitent pas, les algorithmes génétiques, les méthodes évolutionnaire, réseaux de neurones, recherche tabou, recuit simulé, système de colonie de fourmis, scatter search, recherche local,... et leurs hybrids. Le succès de ces méthodes est dû à leur capacité de résoudre quelques problèmes combinatoires difficiles.

Heuristiques

Mot grec, vient du verbe "heuriskein" qui signifie "trouver". L'heuristique est définie par *Reeves* en 1995 [17], c'est une technique qui cherche une bonne solution (presque optimale) avec un coût raisonable sans garantir l'optimalité. Souvent, les heuristiques sont spécifiques au problème considéré, une méthode utilisée pour résoudre un problème ne peut pas être utilisée pour résoudre un problème différent.

Notion de Voisinage

Déterminer le voisinage consiste à caractériser tous ses éléments. Le voisinage est souvent représenté par une fonction N qui, à un point x, associe un ensemble de points $N(x)$. Il existe une infinité de manières de choisir N, il faut adapter ce choix au problème, c'est-à-dire choisir la meilleure fonction N selon le problème considéré.

Définition 11 *Fonction de voisinage* : Soit X l'ensemble de recherche d'un problème. Une fonction de voisinage N est une association $N : X \rightarrow 2^X$, définissant, pour chaque point $x \in X$, un ensemble $N(x) \subseteq X$ de points "proches" de X

3.3 Méta-heuristique pour le mono-objectif

Lorsqu'on résout un problème d'optimisation mono-objectif, on est amené à chercher une seule solution optimale, la meilleure sur l'ensemble de toutes les solutions réalisables. Ceci est atteint par les méthodes de résolution exactes. Les méthodes exactes ont l'avantage de fournir une garantie sur le résultat obtenu,

qui est prouvé comme étant optimal, mais très souvent on ne peut pas garantir de trouver la solution optimale en un temps raisonnable. Pour cela, les méthodes approchées interviennent pour résoudre ces problèmes avec toutes leurs versions linéaire, non linéaire et surtout de grande de taille de données. Elles sont très utiles pour les problèmes d'optimisation, permettent de trouver des solutions approchées pour les instances de très grande taille en un temps raisonnable .

Les méthodes approchées sont basées généralement sur la recherche locale ou les algorithmes évolutionnaires ou hybrides.

Nous n'allons présenter dans cette section que deux importantes méta-heuristiques car nous allons les détailler dans la section suivante pour les problèmes multi-objectif.

3.3.1 Recherche locale

La recherche locale démarre d'une solution initiale et essaie de l'améliorer, en cherchant une solution meilleure dans le voisinage de la solution courante. Le processus de la recherche est réitéré jusqu'à ce qu'aucune amélioration de la solution courante ne pourrait être faite. l'algorithme général de la recherche locale se résume à :

Algorithme 1 Recherche Locale

```
1: Soit une solution initiale S
2: Amelior = vrai,
3: repeat
4:    S'= Meilleur Voisin(X)
5:    if cot(S) > cot(S') then
6:       S= S'
7:    else
8:       Amelior = faux,
9:    end if
10: until Amelior = faux.
11: return
```

3.3.2 Algorithme génétique

Nous donnons ici un schéma général d'un algorithme génétique :

Algorithme 2 Algorithme génétique

1: Générer aléatoirement une population initiale d'individus.
2: Evaluation : Affecter à chaque individu son coût.
3: **repeat**
4: Sélection : Établir une liste de paires d'individus susceptibles de se reproduire, le critère de sélection favorise les meilleurs
5: Reproduction : Appliquer à chaque individu son coût.
6: Evaluation : Affecter à chaque individu sont coût.
7: Remplacement : Produire la nouvelle population en remplaçant les individus de manière à favoriser encore les meilleurs.
8: **until** un certain critère d'arrêt.
9: **return**

Un algorithme génétique repose sur le principe de l'évolution naturelle d'organismes en respectant les phénomènes d'hérédité et la loi de survie énoncée par *Drawin*, selon lesquels, dans une population d'individus, ce sont les plus forts, c'est à dire les mieux adaptés à l'environnement qui survivent et pourront donner une descendance.

3.4 Méta-heuristique pour le multi-objectif

3.4.1 Méta-heuristiques à solution unique

Les méta-heuristiques à solution unique sont basées sur un algorithme de recherche de voisinage qui commence avec une solution initiale, puis l'améliore pas à pas en choisissant une nouvelle solution dans son voisinage. Les plus classiques sont :

Recherche locale

Les méta-heuristiques dites de recherche locale, ou de descente sont très anciennes et doivent leur succès à leur rapidité et leur simplicité. A chaque pas de la recherche, ces méthodes progressent vers une solution voisine de meilleure qualité.

Le principe de base d'une recherche locale est de partir d'une configuration initiale (d'une affectation complète) et par un processus itératif, de remplacer la configuration courante par une meilleure prise dans ce qui est défini comme son voisinage. L'idée est donc d'être capable, si on améliore une configuration progressivement, d'atteindre une solution.

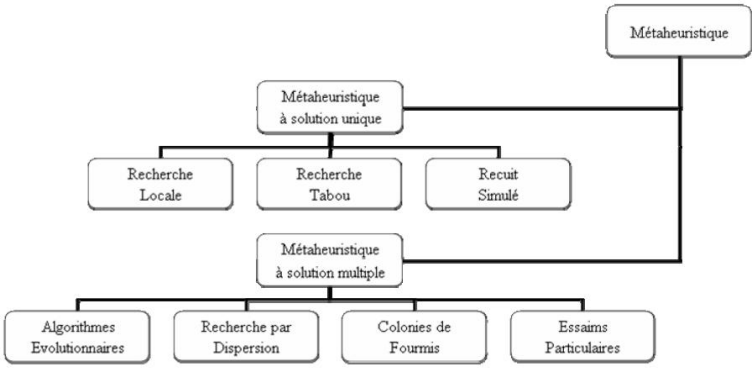

FIG. 3.1 – Méta-heuristiques

Recuit simulé

Le recuit simulé est une technique d'optimisation qui est inspirée des méthodes de simulation de la mécanique statistique dans les année 1950. La méthode de recuit simulé considère une solution initiale et recherche dans son voisinage une autre solution de façon aléatoire. L'originalité de cette méthode est qu'il est possible de se diriger vers une solution voisine de moins bonne qualité avec une probabilité non nulle, ceci permet d'échapper aux optima locaux. Le processus du recuit simulé répète une procédure itérative qui cherche des configurations de coût plus faible tout en acceptant de manière contrôlée des configurations qui dégradent la fonction de coût.

Recherche Tabou

Le mot tabu (ou le taboo) en français tabou, vient de *Tongan*, une langue de polynesia, où il a été employé par les aborigènes de l'île du *Tonga* pour indiquer les choses qui ne peuvent pas être touchées parce qu'elles sont sacrées [22].

La recherche tabou a été introduite par *Glover* en 1948 et a montré ses performances sur de nombreux problèmes d'optimisation. Le principe de l'algorithme est le suivant : à chaque itération, le voisinage de la solution courante est examiné et la meilleure solution est sélectionnée, même si elle est moins bonne que la solution courante. Afin d'éviter le phénomène de cyclage entre plusieurs solutions, la méthode interdit les mouvements aboutissant à une solution récemment visitée. Pour cela, une liste taboue contenant les attributs des dernières solutions visitées est tenue à jour. Chaque nouvelle solution considérée enlève de cette liste la solution la plus anciennement visitée. Ainsi, la recherche de la solution suivante se fait

dans le voisinage de la solution courante sans considérer les solutions appartenant à la liste taboue.

La recherche tabou a changé de façon spectaculaire notre capacité de résoudre des problèmes pratiquement importants. La recherche tabou a fait l'objet de bien des développements ces dernières années. Elle est utilisée pour traiter un grand nombre de problèmes d'optimisation : problème de satisfaisabilité maximum *(Hansen et Jaumard, 1987)*, problème de reconnaissance *(Hertz et Werra, 1987)*, affectation des lignes de telecomunications *(Oliveira et Stourd, 1989)*, problème de coloration dans un graphe *(Hertz et Werra, 1987)*, problème de sac à dos *(Gandibleux et Freville, 2000)* [19, 20].

L'algorithme

La méthode de recherche tabou est décrite dans l'algorithme suivant :

Algorithme 3 Algorithme de Recherche Tabou

```
1: Initialisation : Setbest ← ∅,
2: Iter ← 0,
3: S* ← la solution initiale,
4: while Iter ≤ Itermax do
5:    for chaque Voisin S de S* non tabou do
6:       Calculer F(S)
7:       if ∀i  F_i(S) < F_i(S*) then
8:          Setbest ← S,
9:          (mise à jour de Setbest ),
10:      end if
11:   end for
12:   if Condition d'arrêt ou Iter = Itermax then
13:      Iter ← Itermax,
14:   else
15:      Iter ← Iter + 1,
16:      S* ← S,
17:      (mise à jour de la liste Tabou ),
18:   end if
19: end while
20: return
```

∘ $Setsolution$: l'ensemble des meilleures solutions trouvées.

∘ $Iter$, $Itermax$: le nombre d'iterations en cours et le nombre d'itérations maximum que l'algorithme ne doit pas dépasser.

∘ S^*, S : la meilleure solution de l'itération précédente, et la solution en cours.

3.4.2 Méta-heuristiques à base de population

La deuxième classe de méta-heuristiques est celle des algorithmes évolutionnaires. On peut distinguer trois grandes classes : les algorithmes génétiques, les stratégies d'évolutions et la programmation évolutive. Ces méthodes se distinguent par leur manière de représenter l'information et par leur façon de faire évoluer la population d'une génération à l'autre.

Les algorithmes génétiques

Les algorithmes génétiques on été introduits par *Holland en 1975* [1], comme un modèle de méthode adaptative. Ils s'appuient sur un codage de l'information sous forme de chaînes binaires de longueur fixe et d'un ensemble d'opérateurs génétiques : la sélection, la mutation, le croisement...etc.

Programmation génétique

La programmation génétique consiste à faire une population constituée d'un grand nombre de programmes. La plupart des algorithmes de programmation génétique travaillent avec une population modélisée sous forme d'arbre. Le principe est le même que pour les algorithmes génétiques classiques, mais les opérateurs de mutation et de croisement sont différents.

Colonies de fourmis

Le système de fourmis est initialement inventé par *Dorigo*, basée sur des observations faites sur les fourmis se déplaçant du nid à la source de nourriture et vice-versa. Dans la nature, les fourmis sont capables de trouver le plus court chemin entre une source de nourriture et leur nid, non pas en utilisant des repères visuels, mais en suivant une piste de phéromone. Ce qui a pour effet de créer une piste chimique, les fourmis peuvent sentir ces phéromones qui ont un rôle de marqueurs de chemin : quand les fourmis choisissent leur chemin, elles ont tendance à choisir la piste qui porte la plus forte concentration de phéromone. Ceci est donc applicable à la recherche de bonnes solutions dans un problème d'optimisation.

3.4.3 Autres méthodes heuristiques

Nous avons énuméré les grandes familles de méta-heuristiques pour l'optimisation approchée, basée sur un seul individu, ou sur une population entière. Évidemment, d'autres heuristiques plus spécifiques, ou des classes de méta-heuristiques moins répandues existent dans la littérature.

On trouve de nombreuses heuristiques qui utilisent les particularités du problème traité afin de générer de manière partiellement stochastique ou non, de bonnes solutions. Ces approches sont appelées des *heuristiques constructives*. Parmi ce type d'approche, les algorithmes gloutons sont bien connus. Ils consistent à créer des solutions pas à pas, en exploitant les particularités du problème.

Méthodes gloutonnes

Une méthode gloutonne correspond à une séquence de décisions permettant de construire une solution. A chaque fois qu'une décision est prise à propos de la construction de la solution, cette décision est irrévocable.

Une méthode gloutonne consiste à fixer à chaque étape la valeur d'une variable sans remettre en cause les choix effectués précédemment

3.5 Conclusion

Nous avons présenté dans ce chapitre quelques méthodes d'optimisation issues des méta-heuristiques pour l'optimisation multi-objectif. Ces méthodes ont montré leur efficacité pour trouver des solutions approchées satisfaisantes pour un grand nombre de problèmes. Néanmoins, ces méthodes n'offrent pas de garantie de trouver la solution optimale.

4

Résolution du problème d'enchères Combinatoires Bi-objectifs par une Méthode Hybride

Contents

4.1 Introduction

Dans ce chapitre, nous exposerons la deuxième partie de notre travail comprenant la méthode de résolution du problème d'enchères combinatoires bi-objectif. Avant de parler de la méthode hybride que nous avons construit, nous commencerons par rappeler quelques modes d'hybridations qui existent dans la littérature.

Les méthodes de résolution des problèmes d'optimisation sont très diverses, et sont destinées à leur résolution soit de manière exacte, soit de manière approchée. Les méthodes exactes ont l'avantage de fournir une garantie sur le résultat obtenu, qui est prouvé comme étant optimal, mais très souvent ne garantit pas de trouver la solution optimale en un temps raisonnable. Les heuristiques, quant à elles, permettent de fournir très rapidement des solutions intéressantes mais n'offrent pas de garantie de trouver la solution optimale.

Pour pallier ce sacrifice, une idée largement répandue pour la conception de solveurs plus efficaces et robustes, consiste à combiner plusieurs paradigmes de résolution, afin de bénéficier des atouts respectifs de chacun d'entre eux.

Focacci et al, 2002 dressent un panorama de telles utilisations de recherche locale dans la programmation par contraintes. Nous retrouvons alors des hybridations dans lesquelles la recherche locale est utilisée avec des mécanismes complets pour rendre les voisinages étendus plus intéressants.

Hybridation
C'est l'action de combiner ou de coopérer deux ou plusieurs méthodes d'optimisation (méta/ méta, méta/ exacte, ...etc), le but de cette coopération est de fournir des résultats supérieurs aux deux méthodes qui les composent. Nous allons voir qu'il existe plusieurs manière de combiner ces méthodes selon le niveau, le mode, etc.

4.2 Niveaux d'hybridations

La première étape de la classification consiste à distinguer les méthodes de coopération. On distingue les hybridations de bas niveau et les hybridations de haut niveau.

L'hybridation de bas niveau modifie les éléments fonctionnels qui constituent une méthode d'optimisation. Dans cette classe d'hybridation, une fonction interne d'une méthode d'optimisation est remplacée par une méthode d'optimisation.

Contrairement, l'hybridation de haut niveau conserve l'intégrité des méthodes qu'elle lie. Il n'y a pas de relation directe entre les mécanismes internes des méthodes d'optimisation.

4.3 Modes d'hybridations

Le mode d'hybridation distingue les hybridations en mode *relais* et des hybrides en mode *teamwork* ou aussi Co-évolutionnaire.

En mode relais, les méthodes d'optimisation hybridées opèrent les unes après les autres dans un ordre prédéterminé.
Hormis la première, chaque méthode impliquée dans l'hybridation reçoit en entrée le résultat produit par la précédente, à la manière d'un pipeline.

La classe de recherches hybrides en mode teamwork intègre les modèles d'optimisation hybridés où les méthodes sont hybridées en parallèle. Chaque méthode hybridée conduit une recherche dans un espace de solutions donné.
Très souvent, l'hybridation en mode teamwork fait intervenir une solution d'optimisation à base de population de solutions.

4.3.1 Classes hiérarchiques

Cette classification hiérarchiques aboutit, sur les deux propriétés qui sont le niveau et le mode de l'hybridation à quatre classes d'hybridations :

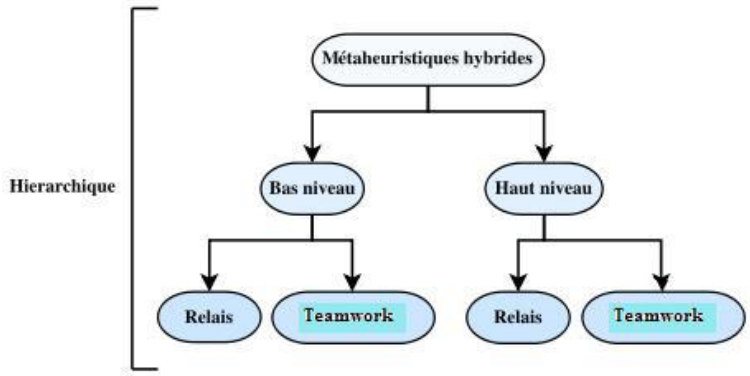

FIG. 4.1 – Les classes hiérarchiques

4.3.2 Classe LRH -low level relay hybrid-

La classe d'hybridation de bas niveau en mode relais (LRH) regroupe les hybridations constituées d'une méthode d'optimisation à solution unique dans laquelle est insérée une autre méthode d'optimisation. Les méta-heuristiques utilisant une population ne sont pas adaptées à ce type d'hybridation en mode relais.

Hybridation méta / méta de type LRH

Martin et al ont réalisé une hybridation de type (LRH) qui combine un recuit simulé et une méthode de descente déterministe pour leur résolution. Le principe général de leur algorithme hybride est de limiter la recherche du recuit simulé à l'espace des optima locaux par l'intégration de la méthode de descente au recuit simulé. L'hybridation se déroule comme suit : elle part d'une solution courante qui est un optimum local, on la perturbe jusqu'à obtenir une solution intermédiaire suffisamment éloignée dans l'espace de recherche. Puis, cette solution intermédiaire est optimisée par une méthode de descente jusqu'à l'obtention d'un minimum local qui devient un candidat du recuit simulé. Enfin, si le candidat est rejeté (suivant plusieurs critères de qualité), on revient à la solution courante, s'elle est acceptée , elle devient la nouvelle solution courante. Cette technique d'hybridation peut facilement conduire à une solution de recherche qui cycle. En effet, on conçoit aisément que la solution intermédiaire et la descente soient basées sur le même opérateur de voisinage, alors la descente risque de retourner à la solution courante.

Hybridation méta / exacte de type LRH

Ce type d'hybridation est plus simple en ayant une méthode heuristique au service d'une méthode exacte. Un exemple de ce type de coopération est proposé par *Augerat et al* [2].

4.3.3 Classe HRH -high level relay hybrid-

Dans l'hybridation de haut niveau en mode relais, les heuristiques et méthodes exactes hybridées conservent leur intégrité. Les méthodes hybrides de types HRH sont exécutées en séquence. Par exemple, on sait que les algorithmes à base de population de solutions ne parviennent pas à ajuster finement les solutions proches des bons optima, on peut améliorer la solution, en l'utilisant comme solution de départ d'un algorithme de recherche locale.

A l'inverse, leur force réside dans la capacité de trouver rapidement des régions de bonne qualité, même pour des espaces de recherche très vastes ou très complexes. Une fois ces régions repérées, il peut être intéressant de poursuivre la recherche en affinant les solutions performantes qui s'y trouvent en appliquant une recherche itérative à solution unique.

Hybridation méta / méta de type HRH

Pour ce type d'hybridation HRH de nombreux auteurs ont conçu des hybridations. On cite l'exemple où les auteurs ont utilisé une recherche tabou optimisé pour une population issue d'un algorithme génétique : donc il s'agit d'une hybridation HRH, impliquant un algorithme génétique suivi d'une recherche tabou, l'étude a montré que l'approche fournie des solutions meilleures que l'algorithme génétique ou la recherche tabou employés seuls.

Hybridation méta / exacte de type HRH

Cette classe d'hybridation inclut notamment les algorithmes exactes utilisant des heuristiques afin d'obtenir les solutions initiales, ou simplement des bornes.

Kliepis et al proposent une hybridation entre Branch and Bound et un algorithme génétique. Dans ce travail, les auteurs alternent des exécutions de Branch and Bound et l'algorithme génétique.

Bent et Van proposent un algorithme hybride en deux phases, en un premier temps l'algorithme minimise le problème étudié par un algorithme de recuit simulé, puis les solutions trouvées sont minimisées par une recherche exacte sur un large voisinage.

4.3.4 Classe LTH -low level Teamwork hybrid-

La classe d'hybridation de bas niveau en mode teamwork (LTH) regroupe les hybridations constituées d'une méthode d'optimisation à population de solutions. Ici les opérateurs sont remplacés par une méthode d'optimisation. Dans cette classe aussi, on peut distinguer deux cas :

- Hybridation méta / méta de type LTH.
- Hybridation méta / exacte de type LTH.

4.3.5 Classe HTH -high level Teamwork hybrid-

L'hybridation de haut niveau teamwork (HTH) implique un ensemble de méta-heuristiques qui travaillent en parallèle et coopèrent pour trouver la solution optimale d'un problème. Dans cette classe aussi, on peut distinguer deux cas :

- Hybridation méta / méta de type HTH.
- Hybridation méta / exacte de type HTH.

4.4 Méthode de résolution

La première partie de notre travail avait porté sur la proposition du modèle d'enchères combinatoires multi-objectif. Dans cette deuxième partie, nous allons construire un algorithme hybride pour résoudre le cas Bi-objectif.

L'hybridation que nous proposons utilise une méthode exacte et une heuristique,(Branch and Bound et Recherche Tabou). Nous rappellerons ici le modèle et donnerons quelques notations utilisées dans l'algorithme :

Modèle

$$\begin{cases} Optimiser\ \ Z(x) = (f_1(x),\ f_2(x)) \\ x \in D \end{cases}$$

$x = (x_1,\ x_2,\ \ldots,\ x_k)$ est le vecteur représentant les variables de décision, $x_i \in \{0,\ 1\}^n, i = 1,\ 2, \ldots,\ n$. D représente l'ensemble des solutions réalisables et chacune des fonctions $f_i(x)$ est à optimiser (minimiser ou maximiser).

Notations

◦ $Setsupp, Setnosupp$:ensemble des solutions pareto supportées et ensemble des solutions non supportées respectivement.

◦ $N(x)$: ensemble de tous les points voisins de x.

◦ $x_{voisinage}$:une solution voisine de l'ensemble de voisinage $N(x)$.

◦ f_1, f_2 : les fonctions objectifs du problème original.

◦ $f = (f_1, f_2)$: fonction objectif construite par l'algorithme, elle s'écrit en fonction de f_1 et f_2 : $f(x) = f_1(x)(y_1 - y_2) + f_2(x)(x_2 - x_1)$, où $x_1,\ x_2$ les solutions trouvées par Branch and bound à chaque itération. $y_1 = f_1(x_1),\ x_2 = f_2(x_2)$

◦ D :l'ensemble de toutes les solutions réalisables.

On a les problèmes :

$$(P_1) \qquad x_1 \in \{x \in D\ /\ f_1(x_1) = \max f_1(x)\},$$

$$(P_2) \qquad x_2 \in \{x \in D\ /\ f_2(x_2) = \max f_2(x)\},$$

$$(P_3) \qquad x' \in \{x \in D\ / f(x') = \max f(x)\ / f = (f_1, f_2)\}$$

La méthode

Voilà, un schéma général de notre algorithme :

Algorithme 4 Multi-Object Auction by Branch and Bound & tabu search

Initialisation : $Setnonsupp \leftarrow \emptyset$, $Setsupp \leftarrow \emptyset$.
Résoudre par **Branch and Bound** :

$$\begin{cases} x_1 \in \{x \in D \ / \ f_1(x_1) = \max f_1(x), \}, \\ x_2 \in \{x \in D \ / \ f_2(x_2) = \max f_2(x), \}, \end{cases}$$

Mise à jour de $Setsupp$.
Appliquer la procédure de **recherche tabou** :
for chaque $x \in Setsupp$ **do**
    Chercher le voisinage $N(x)$,
    for chaque $x_{voisin} \in N(x)$ **do**
        Calculer $f(x_{voisin})$
        if $f_1(x_{voisin}) < f_1(x)$ ou $f_2(x_{voisin}) < f_2(x)$ **then**
            $Setnonsupp \leftarrow x_{voisin}$,
            (mise à jour de $Setnonsupp$),
        end if
    end for
end for
Résoudre par **Branch and bound** $x' \in \{x \in D \ / f(x') = \max f(x) \ / f = (f_1, f_2)\}$,
if $x' \in Setsupp$ **then**
    Arrêter.
else
    $\begin{cases} x_1 \leftarrow x', & \text{aller à } \textbf{Recherche tabou} \text{ ,} \\ x_2 \leftarrow x', & \text{aller à } \textbf{Recherche tabou} \end{cases}$
end if
return

Comme il s'agit de la résolution d'un problème Bi-objectifs, la recherche ne se limite pas à trouver une seule solution optimale mais plutôt à la recherche d'un ensemble de solutions qui donnent le meilleur compromis.

• Notre algorithme commence par résoudre les deux problèmes (P_1) et (P_2) par la méthode de Branch and Bound.
Les solutions trouvées dans cette résolution sont utilisées comme solution de départ pour la recherche tabou, qui de sa part fait une recherche dans le voisinage de ces solutions et si une bonne solution est rencontrée, elle est ajoutée à l'ensemble des solutions efficaces.

• Puis pour chaque nouveau x trouvé par la résolution de (P_3) par la méthode de Branch and Bound, On applique la recherche tabou en démarrant de ce x trouvé pour améliorer l'ensemble des solutions efficaces.

4.5 Exemples

Dans la section précédente nous avons présenté notre algorithme construit par une hybridation des méthodes Branch and Bound et Recherche Tabou, que nous avons implementé en utilisant le langage de programmation **MATLAB Version 7.0**. Notre choix s'est porté sur ce langage car nous estimons qu'il est l'un des langages de programmation mathématique les plus puissants qui existent actuellement.

Dans cette section du chapitre, nous avons choisi de présenter quelques résultats obtenus en exécutant l'algorithme proposé. Pour ce faire, nous allons détaillé le résultats trouvé pour quatre exemples et nous allons les comparer aux mêmes résultats trouvés par différents auteurs.

Les exemples traités dans cette section sont des instances du problème du sac à dos bi-objectif qui sont équivalents au problème d'enchères combinatoires bi-objectif (Voir chapitre Enchères Combinatoires).

4.5.1 Exemple 1

Pour cet exemple on a :

◦ Deux vendeurs $K = 2$, chacun propose un article.
◦ Deux articles $M = \{1,\ 2\}$ et avec 16 unités de chaque article.
◦ Les enchères soumises sont

Enchères	Articles demandés	Nombre d'articles demandés	Prix proposé
b1	(1, 2)	(4, 4)	(11, 9)
b2	(1, 2)	(2, 2)	(5, 2)
b3	(1, 2)	(8, 8)	(7, 16)
b4	(1, 2)	(7, 7)	(13, 5)
b5	(1, 2)	(5, 5)	(3, 4)

Le problème mathématique de cet exemple est :

$$\begin{cases} \max f_1(x) = 11x_{b_1} + 5x_{b_2} + 7x_{b_3} + 13x_{b_4} + 3x_{b_5} \\ \max f_2(x) = 9x_{b_1} + 2x_{b_2} + 16x_{b_3} + 5x_{b_4} + 4x_{b_5} \\ S.c \\ 4x_{b_1} + 2x_{b_2} + 8x_{b_3} + 7x_{b_4} + 5x_{b_5} \leq 16, \\ x_{b_j} \in \{0,\ 1\}, \qquad \forall j \in \{1,\ 2,\ 3,\ 4,\ 5\} \end{cases}$$

Nous avons résolu ce problème par l'algorithme proposé et nous avons obtenu les solutions suivantes :

$$x^1 = (1,\ 1,\ 0,\ 1,\ 0) \text{ avec } (f_1(x^1),\ f_2(x^1)) = (29,\ 16)$$

$$x^2 = (1,\ 1,\ 1,\ 0,\ 0) \text{ avec } (f_1(x^2),\ f_2(x^2)) = (23,\ 27)$$

$$x^3 = (1,\ 0,\ 0,\ 1,\ 1) \text{ avec } (f_1(x^3),\ f_2(x^3)) = (27,\ 18)$$

L'algorithme trouve toutes les solutions efficaces du problème, ces même résultats ont été trouvé par d'autre auteurs *Ehrgott* [14], *Ulungu* [40]. Les deux auteurs résolvent le problème avec un algorithme de Branch and Bound seulement mais chacun son propre algorithme.

4.5.2 Exemple 2

Pour cet exemple on a :

- Deux vendeurs $K = 2$, chacun propose un article.
- Deux articles $M = \{1,\ 2\}$ et avec 15 unités de chaque article.
- Les enchères soumises sont

Enchères	Articles demandés	Nombre d'articles demandés	Prix proposé
b1	(1, 2)	(8, 8)	(6, 12)
b2	(1, 2)	(6, 6)	(4, 10)
b3	(1, 2)	(4, 4)	(4, 5)
b4	(1, 2)	(3, 3)	(4, 3)
b5	(1, 2)	(2, 2)	(3, 1)

Le problème mathématique de cet exemple est :

$$\begin{cases} \max f_1(x) = 6x_{b_1} + 4x_{b_2} + 4x_{b_3} + 4x_{b_4} + 3x_{b_5} \\ \max f_2(x) = 12x_{b_1} + 10x_{b_2} + 5x_{b_3} + 3x_{b_4} + x_{b_5} \\ S.c \\ 8x_{b_1} + 6x_{b_2} + 4x_{b_3} + 3x_{b_4} + 2x_{b_5} \leq 15, \\ x_{b_j} \in \{0,\ 1\}, \qquad \forall j \in \{1,\ 2,\ 3,\ 4,\ 5\} \end{cases}$$

Pour cet exemple nous avons obtenu les solutions suivantes, en résolvant par notre algorithme :

$$x^1 = (0,\ 1,\ 1,\ 1,\ 1) \text{ avec } (f_1(x^1),\ f_2(x^1)) = (15,\ 19)$$

$$x^2 = (1,\ 1,\ 0,\ 0,\ 0) \text{ avec } (f_1(x^2),\ f_2(x^2)) = (10,\ 22)$$
$$x^3 = (1,\ 0,\ 1,\ 1,\ 0) \text{ avec } (f_1(x^3),\ f_2(x^3)) = (14,\ 20)$$

Les auteurs *X. Gandibleux et A. Freville* [19] dans leur article : *Tabu Search Based Procedure for Solving the 0-1 MultiObjective Knapsack Problem : The Two Objectives Case*, résolvent cet exemple en utilisant la recherche tabou seulement et trouvent toutes les solutions efficaces comme dans notre cas.

4.5.3 Exemple 3

Pour cet exemple on a :

- Deux vendeurs $K = 2$, chacun propose un article.
- Deux articles $M = \{1,\ 2\}$ et avec une unité de chaque article.
- Les enchères soumises sont

Enchères	Articles demandés	Nombre d'articles demandés	Prix proposé
b1	(1, 2)	(1, 1)	(6, 1)
b2	(1, 2)	(1, 1)	(3, 3)
b3	(1, 2)	(1, 1)	(1, 6)

Le problème mathématique de cet exemple est :

$$\begin{cases} \max f_1(x) = 6x_{b_1} + 3x_{b_2} + x_{b_3} \\ \max f_2(x) = x_{b_1} + 3x_{b_2} + 6x_{b_3} \\ S.c \\ x_{b_1} + x_{b_2} + x_{b_3} \leq 1, \\ x_{b_j} \in \{0,\ 1\}, \quad \forall j \in \{1,\ 2,\ 3\} \end{cases}$$

Pour cet exemple, nous avons obtenu les solutions suivantes, en résolvant par notre algorithme :

$$x^1 = (1,\ 0,\ 0) \text{ avec } (f_1(x^1),\ f_2(x^1)) = (6,\ 1)$$
$$x^2 = (0,\ 1,\ 0) \text{ avec } (f_1(x^2),\ f_2(x^2)) = (3,\ 3)$$
$$x^3 = (0,\ 0,\ 1) \text{ avec } (f_1(x^3),\ f_2(x^3)) = (1,\ 6)$$

L'algorithme trouve toutes les solutions efficaces du problème, ces même résultats ont été atteints par *Ulungu* [40]. IL résout le problème avec un algorithme de Branch and Bound seulement.

4.5.4 Exemple 4

Pour cet exemple on a :

◦ Deux vendeurs $K = 2$, chacun propose un article.
◦ Deux articles $M = \{1,\ 2\}$ et avec 20 unités de chaque article.
◦ Les enchères soumises sont

Enchères	Articles demandés	Nombre d'articles demandés	Prix proposé
b1	(1, 2)	(4, 4)	(11, 9)
b2	(1, 2)	(3, 3)	(4, 8)
b3	(1, 2)	(2, 2)	(5, 2)
b4	(1, 2)	(1, 1)	(1, 2)
b5	(1, 2)	(8, 8)	(7, 16)
b6	(1, 2)	(7, 7)	(13, 5)
b7	(1, 2)	(6, 6)	(4, 9)
b8	(1, 2)	(5, 5)	(3, 4)

Le problème mathématique de cet exemple est :

$$\begin{cases} \max f_1(x) = 11x_{b_1} + 4x_{b_2} + 5x_{b_3} + x_{b_4} + 7x_{b_5} + 13x_{b_6} + 4x_{b_7} + 3x_{b_8} \\ \max f_2(x) = 9x_{b_1} + 8x_{b_2} + 2x_{b_3} + 2x_{b_4} + 16x_{b_5} + 5x_{b_6} + 9x_{b_7} + 4x_{b_8} \\ S.c \\ 4x_{b_1} + 3x_{b_2} + 2x_{b_3} + x_{b_4} + 8x_{b_5} + 7x_{b_6} + 6x_{b_7} + 5x_{b_8} \leq 20, \\ x_{b_j} \in \{0,\ 1\}, \qquad \forall j \in \{1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8\} \end{cases}$$

Pour cet exemple, nous avons trouvé les solutions efficaces suivantes :

$$x^1 = (1,\ 0,\ 1,\ 1,\ 0,\ 1,\ 1,\ 0) \text{ avec } (f_1(x^1),\ f_2(x^1)) = (34,\ 27)$$
$$x^2 = (1,\ 1,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0) \text{ avec } (f_1(x^2),\ f_2(x^2)) = (28,\ 37)$$

Mais dans ce cas, notre algorithme trouve seulement quelques solutions efficaces. Tandis que dans [40] toutes les solutions efficaces sont trouvées, l'auteur dans cet article résout ce problème avec un algorithme exact Branch and Bound.

Discussion

Sur les quatre exemples que nous avons exposés dans cette partie, notre algorithme trouve toutes les solutions efficaces pour trois exemples 1, 2, 3 , mais il échoue sur les solutions du quatrième exemple. Dans cet algorithme, on commence la recherche par l'application de l'algorithme Branch and Bound pour trouver quelques solutions efficaces supportées, puis on applique la recherche tabou à partir de la solution donné par la procédure de Branch and Bound, pour compléter l'ensemble des solutions efficaces (supportées et non supportées).

Mais nous savons que la recherche tabou est une heuristique qui ne garantit pas toujours de trouver la solution optimale. Ce qui fait que sur l'exemple 4 nous n'obtenons pas toutes les solutions efficaces.

4.6 Conclusion

Les méthodes d'optimisation sont divisées en deux grandes classes, selon qu'elles résolvent les problèmes de manière exacte ou approchée. Au fur et à mesure des années et de l'augmentation exponentielle de puissance de calcul des ordinateurs, ces méthodes d'optimisation sont devenues de plus en plus évoluées.

Cette puissance de calcul disponible a abouti depuis quelques années à de nombreux travaux tentant de faire hybrider plusieurs de ces méthodes d'optimisation entre elles. Dans un premier temps l'hybridation était réalisée entre méthodes d'optimisation approchées. Mais actuellement, de plus en plus des hybridations entre méthodes exactes et approchées voient le jour.

Conclusion Générale

Dans ce travail, nous avons abordé le domaine de la coopération entre méthodes d'optimisation dans le cadre de la résolution de problèmes multi-objectif, difficiles. Afin d'évaluer les différents mécanismes de coopération, nous avons réalisé notre expérimentation sur le problème d'enchères combinatoires bi-objectif. Nous avons expérimenté notre méthode sur un problème multi-objectif. Les problèmes multi-objectifs sont connus pour être particulièrement difficiles, ce qui pousse à utiliser des méthodes d'optimisation complexes et hybridés pour leur résolution.

Dans un premier temps, nous avons donc proposé un modèle mathématique qui généralise toutes les formulations existante du problème d'enchères combinatoires.

La principale conclusion que nous pouvons tirer est qu'il existe beaucoup d'opportunités pour développer de tels algorithmes hybridant méthodes exactes et techniques de recherche locale. Un certain nombre d'approches a été présenté, parfois complexes, pouvant être améliorées et étendues vers des applications différentes de celles proposées initialement. Ces techniques partagent néanmoins une philosophie commune sur la combinaison de méthodes exactes et approchées. Une méthode est désignée comme le processus de recherche principal et une autre secondaire est utilisée comme heuristique d'amélioration avec une organisation hiérarchique.

Comme perspectives de recherche, nous proposons de :
- Tester l'algorithme proposé sur des bases de données de grande taille.
- Faire une extension de l'étude au cas des problèmes à trois objectifs et plus.

Bibliographie

[1] BARICHARD, V. *Approches Hybrides pour les Problèmes Multi-Objectifs*. PhD thesis, Université d'Angers, Novembre 2003.

[2] BASSEUR, M. *Conception d'Algorithmes Coopératifs pour l'Optimisation Multi-Objectif : Application aux problèmes d'Ordaonnancement de Type Flow-Shop*. PhD thesis, Université des Sciences et Technologie de Lille, U.F.R D'I.E.E.A., Juin 2005.

[3] BASSEUR, M., TALBI, E.-G., NEBERO, A., AND ALBA, E. Metaheuristics for multiobjective combinatorial optimization problems : Review and recent issues.

[4] BOUGHACI, D. *Approches Methaheuristiques pour les Problèmes Difficiles (MAX-SAT et PDG)*. PhD thesis, Université des Sciences et Technologie de Houari Boumedienne, Février 2008.

[5] CHAIB-DRAA, B. Enchères-négociation. Tech. rep., Université Laval, 2008.

[6] CHAKHAR, S. *Cartographie Décisionnelle Multicritère : Formalisation et Implémentation Informatique*. PhD thesis, Université Paris Dauphine, D.F.R. Sciences Des Organisations, Aout 2006.

[7] DE BORNIER, J. M. Les mécanismes d'enchère.

[8] DE VRIES, S., SCHUMMER, J., AND VOHRA, R. V. On ascending vickrey auctions for heterogeneous objects. *Journal of economic theory, Elseiver* (July 2005).

[9] DE VRIES, S., AND VOHRAY, R. Combinatorial auctions : A survey. Tech. rep., Department of Managerial Economics and Decision Sciences, Kellogg School of Management, Northwestern University, 2000.

[10] DELORME, X., GANDIBLEUX, X., AND DEGOUTIN, F. Résolution approchée du problème de set packing bi-objectifs. Tech. rep., Université de Valenciennes, 2005.

[11] DHAENENS-FLIPO, C. *Optimisation Combinatoire Multi-Objectif : Apport des Méthodes Coopératives et Contribution à l'Extraction de Connaissances*. PhD thesis, Université des Sciences et Technologie de Lille, U.F.R D'I.E.E.A., Octobre 2005.

[12] DOERNER, K. F., GENDREAU, M., GREISTORFER, P., GUTJAHR, W. J., HARLT, R. F., AND REIMANN, M. *Metaheuristics : Progress in Complex Systems Optimization*. Springer, 2007.

[13] DRÉO, J., PÉTROWSKI, A., SIARRY, P., AND TAILLARD, E. *Metaheuristics for Hard Optimization : Simulated annealing, Tabu search, Evolutionary and Genetic Algorithms...* Springer, 2005.

[14] EHRGOTT, M. *Multicriteria Optimisation*. Springer, 2005.

[15] EHRGOTT, M., AND GANDIBLEUX, X. An annotated bibliography of multiobjective combinatorial optimisation.

[16] EHRGOTT, M., AND GANDIBLEUX, X. *Multiple Criteria Optimisation : State of the Art Annotated Bibliographic Surveys*. Kluwer's International Series, 2003.

[17] EHRGOTT, M., AND GANDIBLEUX, X. Approximative solution methods for multiobjective combinatorial optimization. *Socied de Estadistica e Investigacion Operativa* (2004).

[18] ESCUDERO, L. F., LANDETE, M., AND MARIN, A. A branch-and-cut algorithm for the winner determination problem. *Decision Support Systems* (juillet 2008).

[19] GANDIBLEUX, X., AND FREVILLE, A. Tabu search based procedure for solving the 0-1 multiobjective knapsack problem : The two objectives case. *Journal of Heuristics* (2000).

[20] GLOVER, F. Tabou search : A tutorial. Tech. rep., Center for applied Artificiel intelligent Université of Colorado, Aout 1990.

[21] GLOVER, F., AND KOCHENBERGER, G. A. *Handbook of Metaheuristics*. Kluwer's International Series, 2003.

[22] GLOVER, F., AND LAGUNA, M. *TABU SEARCH*.

[23] GONEN, R., AND LEHMANN, D. Linear programming helps solving large multi-unit combinatorial auctions.

[24] GONZALEZ, T. F. *Handbook of Approximation Algorithms and Metaheuristics*. Chapman and Hall/CRC, 2007.

[25] GUO, Y., LIN, A., RODRIGUES, B., AND ZHU, Y. Heuristics for a brokering set packing problem. *Elseiver* (Aout 2006).

[26] IBARAKI, T., NONOBE, K., AND YAGIURA, M. *Metaheuristics : Progress as Real Problem Solvers*. Springer, 2005.

[27] KLEMPERER, P. *Auctions : Theory and practice*. 2002.

[28] KWANG, L., AND SHARKAWI, M. A. E. *Modern Heuristics Optimization Techniques : Theory and Applications to Power Systems*. IEEE Press, 2008.

[29] LAMBERT, T. *Hybridation de méthodes complètes et incomplétes pour la résolution de CSP*. PhD thesis, Université de Nantes, Octobre 2006.

[30] LEHMANN, D., MULLER, R., AND SANDHOLM, T. The winner determination problem,. 2000.

[31] LEYTON-BROWN, K. Resource allocation in competitive multiagent systems. Aout 2003.

[32] LEYTON-BROWN, K., SHOHAM, Y., AND TENNENHOLTZ, M. An algorithm for multi-unit combinatorial auctions. *American Association for Artificial Intelligence* (2000).

[33] PFEIFFER, J. *The Winner Derterminationn Problem in Multi-Unit Combanotorial and The Multi-Dimensional Knapsack Problem, An Anaysis of Optimisation Methods*. PhD thesis, Université Mannheim, Octobre 2006.

[34] RON HOLZMAN, A. N. K.-D., MONDERER, D., AND TENNENHOLTZ, M. Bundling equilibrium in combinatorial auctions. *Games and Economics Behavior, Elseiver* (juin 2002).

[35] SANDHOLM, T., AND SURI, S. Bob : Improved winner determination in combinatorial auctions and generalizations. *Journal of Artificial Intelligence* (Novembre 2001).

[36] SANDHOLM, T., SURI, S., GILPIN, A., AND LEVINE, D. Cabob : A fast optimal algorithm for combinatorial auctions. *Journal of Artificial Intelligence* (2005).

[37] SIARRY, P., AND MICHALEWICS, Z. *Advances in Metaheuristics for Hard Optimization*. Springer, 2008.

[38] STEINER, S. *Computing All Efficient Solutions of Multiobjective Combinatorial Optimisation Problems*. PhD thesis, Department of Computer Science King's College London, Juin 2006.

[39] TORN, A., AND ZILINSKAS, J. *Models and Algorithms for Global Optimization*. Springer, 2007.

[40] ULUNGU, E. L. *Optimisation Combinatoire Multicritére : Détermination de l'Eensemble des Solutions Efficaces et Méthodes Itératives*. PhD thesis, 1997.

[41] VISÉE, M., TEGHEM, J., PIRLOT, M., AND ULUNGU, E. Two-phases method and branch and bound procedures to solve the bi–objective knapsack problem. *Journal of Global Optimization* (1998).