N^o d'ordre : 03/2010 – D/MT

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université des Sciences et de la Technologie Houari Boumediene
Faculté de Mathématiques

Thèse

présentée pour l'obtention du diplôme de Doctorat en Mathématiques
Spécialité Recherche Opérationnelle

par ADICHE Chahrazad ép HECHICHE

Thème

Approches déterministe et stochastique de résolution de problèmes d'optimisation

Soutenue publiquement, le 15 Décembre 2009, devant le jury composé de :

M. MOULAI, Professeur, USTHB	Président
M. AÏDER, Professeur, USTHB	Directeur de Thèse
M.S. AÏDENE, Professeur, UMMTO	Examinateur
S. BOUROUBI, Professeur, USTHB	Examinateur
A. BOUKRA, Maître de Conférences classe A, USTHB	Examinateur
D. CHAABANE, Maître de Conférence classe A, USTHB	Examinateur

Remerciements

Tout d'abord, je remercie ALLAH le tout puissant, de m'avoir aidée dans les moments les plus difficiles. Le parcours n'était pas facile et l'aboutissement n'était pas aussi garanti sans son aide et son assistance. Qu'il soit toujours présent dans mon coeur.

Mes remerciements les plus sincères s'adressent ensuite à mon Directeur de Thèse Monsieur Méziane AÏDER, Professeur à l'Université des Sciences et de la Technologie Houari Boumediene pour avoir accepté le suivi et l'encadrement de cette thèse. Son savoir scientifique et méthodique ainsi que ses remarques et suggestions judicieuses ont apporté beaucoup d'éclaircissements. Je le remercie également pour m'avoir initiée dans le domaine de la recherche et avoir suivi ma progression pas à pas, son esprit est bien marqué dans ce travail. Qu'il trouve à travers ces lignes l'expression de ma profonde et sincère reconnaissance. Car au-delà des mots et des phrases, aucune parole ne saurait exprimer l'infinie gratitude que je lui dois.

Je tiens ensuite à remercier Monsieur Mustapha MOULAI, Professeur à l'Université des Sciences et de la Technologie Houari Boumediene, pour l'honneur qu'il me fait en acceptant de présider ce jury.

Un remerciement particulier à Monsieur Abdelmadjid BOUKRA, Maître de Conférence à l'USTHB, qui même dans un moment très difficile a bien voulu examiner ce travail.

Mes remerciements vont également à Monsieur Mohamed Said AÏDENE, Professeur à UMMTO, Monsieur Sadek BOUROUBI, Professeur à l'USTHB, Monsieur Djamel CHAABANE, Maître de conférence à l'Université des Sciences et de la Technologie Houari Boumediene, pour avoir accepté d'examiner ce travail et participer au jury.

A mes parents, dont l'amour et les sacrifices n'ont pas cessé de combler ma vie

A mon époux

A mes enfants

A mes frères et soeurs

A tous ceux qui me sont chèrs...

Je dédie ce modeste travail

Table des matières

Table des figures

Liste des tableaux

Liste des algorithmes

Introduction

L'Optimisation Combinatoire (OC) occupe une place très importante en recherche opérationnelle, en mathématique discrète et en informatique. Elle consiste à trouver "le meilleur" entre un nombre fini de choix. Autrement dit, à optimiser une fonction avec ou sans contraintes, sur un ensemble fini.

Les problèmes d'optimisation rencontrés en pratique sont rarement mono-objectif. En effet, souvent, deux ou plusieurs objectifs (parfois conflictuels) sont présents (parfois les objectifs sont incommensurables). Il n'existe pas, dans ce cas, un optimum unique ; on cherche, en revanche, un ensemble de solutions "optimales au sens de Pareto", qui forment la "surface de compromis" du problème considéré. L'optimisation multi-objectif [17] s'intéresse à la résolution de ce type de problèmes. Ses racines datent du 19-ième siècle grâce aux travaux en économie de Edgeworth et Pareto [61].

Les travaux de recherche réalisés dans le cadre de la préparation de notre thèse de Doctorat, s'inscrivent dans la thématique de l'Optimisation Combinatoire Multi-Objectif (OCMO). Cette dernière constitue une classe particulière des problèmes d'optimisation muli-objectif. De plus ces problèmes peuvent être modélisés comme des programmes en nombre entiers avec plusieurs objectifs.

Certains problèmes classiques d'optimisation combinatoire mono-objectif, ont été étudiés dans une version multi-objectif [20] : sac à dos [6], ordonnancement [57], plus court chemin dans un graphe [94], arbre recouvrant (minimum spanning tree) [95], affectation [86], voyageur de commerce [70], routage de véhicules [62] etc... et la plupart des benchmarks utilisés dans la comparaison des algorithmes élaborés, ont été réalisés sur des problèmes académiques. Néanmoins, il existe des applications industrielles, portant sur l'optimisation combinatoire multi-objectif, dans différents domaines. Nous pouvons citer parmi celles-ci :

- design de systèmes dans les sciences d'ingénieurs : ailes d'avions (Obayashi et al., 1998), moteurs d'automobiles (Fujita et al., 1998) ;
- finances : investissements financiers, etc ;
- ordonnancement et affectation : ordonancement en productique (Shaw and Fleming, 1996), localisation d'usines, planification de trajectoires de robots mobiles (Fujimara, 1996), etc ;
- agronomie : programme de production agrécole, etc ;
- transport : gestion de containers (Todd and Sen, 1997), design de réseaux de transport (Friesz et al., 1993), tracé autoroutier, etc ;
- environnement : gestion de la qualité de l'air (Loughlin and Ranjithan, 1997), distribution de l'eau (Halhal et al., 1997), ect ;
- télécommunications : design d'antennes (Veldhuizen et al., 1997), affectation de fréquences (Dahl et al., 1995)(Weinberg, 2001), etc.
- télécommunications cellulaires : radiotéléphonie mobile (Meunier et Talbi, 2002).

Dans la littérature, une attention particulère a été porté sur les problèmes à deux objectifs. Ehrgot et Gandibleux [25] présentent une bibliographie sur les publications réalisées en OCMO couvrants plusieurs méthodes différentes et problèmes. Dans ce survey, il est intéressant d'observer comment des problèmes d'OCMO pratiques sont souvent résolus avec la méthode du goal programming [42] et [51]. Il a été noter également, que peu d'articles ont été recensés dans la littérature concernant l'application du branch and bound à l'OCMO (problème du sac à dos [89], [87], [93] et problème de max-ordering shortest path [65]).

Pour des problèmes à plus de deux objectifs et de grandes tailles, il n'existe pas de procedures exactes efficaces, étant donné les difficultés simultanées de la NP-complétude, et du cadre multi-objectif de ces problèmes. Dans ce cas, il semble naturel d'investiguer les possibilités d'application des métaheuristiques pour ces problèmes particulièrement complexes et difficiles. Plusieurs adaptations de métaheuristiques (recuit simulé, recherche tabou, algorithmes évolutionnaires, algorithmes de colonies de fourmis, réseaux de neurones, essaims particulaires,...) ont été proposées dans la littérature pour la résolution des problèmes d'optimisation multi-objectif et la détermination d'une approximation satisfaisante d'une solution, de plusieurs ou de toutes les solutions Pareto optimales, ou en d'autre termes une approximation partielle ou complète de l'ensemble non-dominés.

Très récemment, des algorithmes hybrides combinant la recherche génétique et la re-

cherche locale (limitée à des méthodes de descente) tels que "multiple objective genetic local search" (MOGLS) [44], "genetic tabu search " [12] ainsi que des approches hybrides parallèles, [45] ont été proposées.

Cette thèse porte sur la résolution du problème d'affectation multi-objectif. Tout comme la plupart des problèmes intéressants, il est NP-complet [69], même pour seulement deux objectifs.

La plupart des approches exactes développées dans la littérature, pour la résolution du problème d'affectation bi-objectif, sont basées sur la méthode des deux phases qui utilise les méthodes mono-objectif et les propriétés de dualité du problème d'affectation. La méthode des deux phases a été introduite par Ulungu [85] qui a montré que les solutions efficaces ne sont pas connectées par les pivots du simplexe, contrairement à ce qui a été initialement supposé par Bhatia Malhotra et Puri [15] et que des solutions efficaces non supportées existent. Des résultats expérimentaux [23], basés sur la méthode de dichotomie et utilisant un solveur MIP, ont permis de détecter d'avantage de solutions efficaces.

Plusieurs approches de la méthode des deux phases ont été proposées dans la littérature. Elles diffèrent principalement, dans leurs manières de calculer, dans la deuxième phase, les solutions efficaces non supportées. Dans [63], Przybylski et al.(2008) calculent les solutions efficaces non supportées par une méthode de ranking et donnent également une description complète de la méthode des deux phases pour le problème d'affectation bi-objectif. Cependant, à notre connaissance, aucune des méthodes existantes ne permet de résoudre le problème au delà de deux objectifs. Dans les deux dernières décennies, l'application des techniques métaheuristiques pour le problème ont été introduites (voir [88] et [31]).

Ce document est organisé en cinq chapitres :

Au chapitre un, nous rappelons certaines généralités sur l'optimisation combinatoire multi-objectif, incluant quelques concepts fondamentaux tels que la dominance, la surface de compromis, l'aspect complexité des problèmes d'Optimisation Combinatoire Multi Objectif (OCMO) ainsi que les principales techniques de résolution de ces problèmes.

Au chapitre deux, nous présentons une adaptation de la méthode branch and bound au contexte multi-objectif (MOBB) [2], pour la résolution du problème d'affectation multi-objectif avec trois objectifs et plus.

Au chapitre trois, nous nous proposons d'étudier l'impact de la méthode appliquée pour l'évaluation et la comparaison entre les solutions (agrégation ou relation de dominance) lorsque la recherche locale est utilisée pour explorer l'espace des objectifs du problème d'affectation multi-objectif. Ainsi, deux nouvelles variantes de l'adaptation de la méthode du recuit simulé au contexte multi-objectif (DUSA et DCSA) seront présentées et comparées à l'approche traditionnelle d'Ulungu basée sur l'agrégation.

Au chapitre quatre, nous développons, pour la résolution du problème d'affectation multi-objectif, une approche hybride qui combine la méthode exacte et la méthode approximative présentées aux chapitres deux et trois respectivement.

Enfin, au chapitre cinq, nous traitons la problématique liée à la modélisation des préférences, et au choix de solutions au sein d'un ensemble de compromis. Une méthode de surclassement [5] basée sur une relation de dominance floue, sur l'ensemble des directions de recherche, pour la comparaison et le classement complet des solutions est présentée.

Chapitre 1

Optimisation Combinatoire Multi-Objectif : Généralités

1.1 Introduction

Les problèmes d'optimisation combinatoire sont caractérisés par le fait qu'il possèdent un ensemble fini de solutions réalisables. Cependant la finitude de cet ensemble peut mener à des situations où la détermination d'une solution est une tâche fastidieuse car elle nécessite l'examen d'un nombre très élevé (prohibitif) de situations. Ceci a un impact significatif sur la manière de traiter ces problèmes aussi bien sur le plan théorique que celui de la pratique. Dans ce chapitre nous présentons des généralités sur l'Optimisation Combinatoire Multi-Objectif (OCMO) incluant quelques concepts de base relatifs au context multi-objectif, l'aspect complexité des problèmes d'OCMO ainsi que les principales techniques de résolution de ces problèmes.

1.2 Définitions et concepts de base

Un problème d'OCMO peut souvent être modélisé en un programme en nombre entiers avec des objectifs multiples (i.e., un programme en nombre entiers multi-objectif) de la manière suivante :

$$(POCMO) \begin{cases} \text{“}min\text{”}\ F(x) = (f^1(x), f^2(x), ..., f^m(x)) \\ s.c \\ x \in S \end{cases}$$

Où :

- $m \geq 2$ est le nombre de fonctions objectif,
- $x = (x_1, x_2, ..., x_n)$ est un vecteur de variables entières représentant les variables de décision,
- S est un ensemble discret représentant l'ensemble de solutions réalisables associé à des contraintes d'égalités et/ou d'inégalités (espace de décision),
- $F(x) = (f_1(x), f_2(x), ..., f_m(x))$ est le vecteur des critères à optimiser.

Dans le cadre de l'optimisation multi-objectif, le plus souvent le décideur raisonne plutôt en termes d'évaluation d'une solution sur chaque objectif et se place naturellement dans l'espace des objectifs. Soit Z l'image de S dans l'espace des objectifs.

$$z \in Z \Longleftrightarrow \exists x \in S : F(x) = z \tag{1.1}$$

(POCMO) peut s'écrire alors :

$$(POCMO)' \begin{cases} \text{“}min\text{”}\ z = (z^1, z^2, ..., z^m) \\ s.c \\ z \in Z \end{cases}$$

La cardinalité de l'ensemble Z ne peut jamais dépasser celle de l'ensemble S. L'ensemble Z est fini du moment que S l'est. Une relation d'ordre partiel, appelée relation de dominance, est définie sur l'ensemble Z. Il n'existe, en général, pas de solution meilleure que toutes les autres simultanément pour tous les objectifs. Pour cela, le concept de solution efficace, appelée également solution non dominée et solution Pareto optimale, a été introduit. La définition de solution efficace découle directement de la notion de dominance. Plusieurs relations de dominance ont été définit dans la littérature : la a-dominance [60], la dominance au sens de Geoffrion [25], la cône-dominance [17], . . . Mais la plus célèbre et la plus utilisée reste la dominance au sens de Pareto.

Définition 1.1. Un point $z = (z^1, z^2, \ldots, z^m)$ “domine” un point $w = (w^1, w^2, \ldots, w^m)$ et nous notons $z \preceq w$ si et seulement si pour tout $i \in \{1, \ldots, m\}$, $z^i \leq w^i$ avec au moins un $j \in \{1, \ldots, m\}$ tel que $z^j < w^j$.

La figure 1.1 illustre la notion de dominance au sens de Pareto en dimension 2.

Définition 1.2. Une solution $x^* \in S$ est dite “efficace” (Pareto optimale) pour $(POCMO)$ si et seulement si il n'existe aucune autre solution réalisable $x \in S$, tel que $F(x) \preceq F(x^*)$. Le point $F(x^*)$ est alors appelé point non-dominé.

Il est clair qu'en général, il n'y a pas une seule solution efficace, mais tout un ensemble.

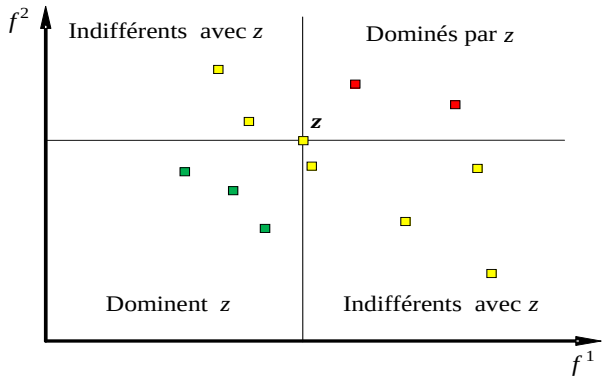


FIG. 1.1: Domination au sens Pareto dans l'espace bi-objectif.

Définition 1.3. Ensemble des solutions non dominées. L'ensemble des solutions non dominées de S, est défini par l'ensemble E : $E = \{x \in S : \nexists y \in S : y \preceq x\}$

De manière analogue, nous pouvons définir la surface de compromis de Z :

Définition 1.4. Front Pareto. Le Front Pareto, appelé aussi surface de compromis, $ND(Z)$ de Z est définie comme suit : $ND(Z) = \{y \in Z/\nexists z \in Z : z \preceq y\}$

Définition 1.5. Une solution réalisable $x*$ est dite "faiblement efficace", si et seulement si il n'existe pas une autre solution $x \in S$ tel que : $f^k(x) < f^k(x^*)$ pour tout $k = 1..m$.

La faible efficacité est une version plus générale de la définition standard de l'efficacité. En effet, l'ensemble de toutes les solutions efficaces est un sous-ensemble de l'ensemble de toutes les solutions faiblement efficaces.

Définition 1.6. Une solution réalisable $x*$ est dite "localement efficace", si et seulement si il existe un scalaire $\delta > 0$ tel que $x*$ est efficace dans $S \cap B(x, \delta)$, où $B(x, \delta) = \{y \in S/\|x - y\| \leq \delta\}$. Où $\|.\|$ désigne une norme quelconque de $\Re^n$.

Définition 1.7. Une solution réalisable $x*$ est dite "proprement efficace", si $x*$ est efficace et si il existe un scalaire $M > 0$ tel que pour tout $k \in \{1, 2, ..., m\}$ et $y \in S$ avec $f^k(y) < f^k(x^*)$, il existe au moins un $j \in \{1, 2, ..., m\}$ $(j \neq k)$ tel que :
$f^j(y) > f^j(x^*)$ et $\frac{f^k(x^*) - f^k(y)}{f^j(y) - f^j(x^*)} \leq M$.

Une particularité des problèmes d'OCMO est que la distinction entre solutions *efficace* et *proprement efficace* disparaît. En effet, comme Z est un ensemble fini et toutes les valeurs des fonctions objectifs sont entières, il y a un nombre fini de valeurs entières pour chaque objectif. Ainsi, le dénominateur n'est jamais inférieur à un et le ratio $\frac{f^k(x^*) - f^k(y)}{f^j(y) - f^j(x^*)}$ est toujours borné. Donc, M existe.

Définition 1.8. Point Ideal. Les coordonnées du "point idéal" $\overline{\pi}$ correspondent aux meilleurs valeurs de chaque objectif des points du front Pareto. Les coordonnées de ce point correspondent aussi aux valeurs obtenues en optimisant chaque fonction objectif séparément. $\overline{\pi}_i = \min\{y_i / y \in ND(Z)\}$.

Définition 1.9. Point Nadir. Les coordonnées du "point Nadir "$\underline{\pi}$ correspondent aux pires valeurs de chaque objectif des points du front Pareto. $\underline{\pi}_i = \max\{y_i / y \in ND(Z)\}$.

le point idéal et le point nadir sont calculés à partir du front Pareto. Le point idéal (resp. le point nadir) domine (resp. est dominé par) tous les autres points de la surface de compromis. Bien que ces points ne soient pas forcément compris dans la zone réalisable, ils servent souvent de pôle d'attraction (resp. de répulsion) lors de la résolution du problème.

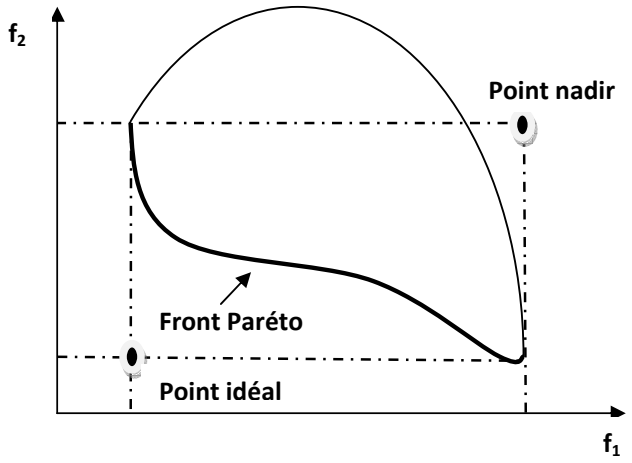


FIG. 1.2: Point idéal et point nadir

La figure 1.2 montre également que le front Pareto peut avoir des propriétés particulières quant à sa forme. Une des principales caractéristiques liée à la forme du front Pareto est la convexité.

Définition 1.10. Convexité. Un ensemble A est convexe, si et seulement si l'équivalence suivante est vérifiée : $x \in A \wedge y \in A \Longleftrightarrow Segment(x, y) \subset A$

La convexité est le premier indicateur de la difficulté d'un problème, parmi d'autres indicateurs tout aussi importants, tels que la continuité, la multi-modalité, la nature des variables de décision (entières ou réelles), . . .

1.3 Complexité et difficultés de l'OCMO

Les problèmes d'optimisation combinatoire multi-objectif sont très complexes. En effet, à la difficulté des problèmes d'optimisation combinatoire due au phénomème d'explosion combinatoire s'additionnent de nouvelles difficultés propres aux problèmes d'optimisation multi-objectifs. Un problème d'OCMO est un problème Non Polynomial (NP) si la détermination d'un point non dominé ne peut être faite en temps polynomial pour toutes les instances du problème. Serafini (1987) a proposé la définition suivante de la NP-complétude pour les problèmes d'OCMO :

Définition 1.11. Pour chaque point $z \in Z$ (du problème (POCMO)') existe-t-il une solution $x \in S$ (du problème (POCMO)) tel : que $F(x) \preceq z$?

D'après la définition 1.11, un problème d'OCMO serait NP si les problèmes d'optimisation combinatoire mono-objectifs qui lui correspondent sont NP. Serafini démontre aussi comment certains problèmes d'optimisation mono-objectif "facile" deviennent NP-complet dans le cas bi-objectif. C'est le cas pour le problème d'affectation et le problème du plus court chemin. Les difficultés rencontrées en OCMO sont relatives d'une part à la localisation de la surface du compromis, dont la forme est souvent méconnue, particulièrement si le problème possède beaucoup de contraintes et d'autre part à la frontière des solutions non dominées lorsque celle ci ne possède pas des propriétés structurelles particulières, telles que la convexité et la continuité. Il est connu depuis longtemps [35], que pour les problèmes de la programmation linéaire multi-objectif :

$$(PLMO)\begin{cases} \text{''}min\text{''}\ F(x) = (f^1(x), f^2(x), ..., f^m(x)) \\ \qquad\qquad\qquad\qquad\qquad\qquad s.c \\ \qquad\qquad\qquad\qquad\qquad\qquad Ax = b, x \geq 0 \end{cases}$$

l'ensemble des solutions efficaces est exactement l'ensemble des solutions obtenues par la résolution des programmes linéaires paramétriques suivants :

$$(PL_\lambda)\begin{cases} \min \sum_{k=1}^{m} \lambda_k f^k(x) \\ \qquad\qquad\qquad s.c \\ \qquad Ax = b, x \geq 0 \end{cases}$$

Où $\lambda \in \Lambda$, avec $\Lambda = \{\lambda \in \Re^m \mid \sum_{k=1}^{m} \lambda_k = 1; \lambda_k > 0; k = 1..m\}$

Théorème 1.1. ***Geoffrion [35]***. *Soit le problème paramétrique* (PL_λ) *avec* $\lambda \in \Lambda$.

a) Si x est solution optimale de (PL_λ), x est solution efficace.

b) Si x est solution efficace et que Z est un ensemble convexe, il existe $\lambda \in \Lambda$ tel que x est solution optimale de (PL_λ).

Remarque 1.1.
- *Si les poids λ_k n'étaient pas tous strictement positifs, la condition a) n'assurerait que la faible efficacité de x dans le cas où x n'est pas solution optimale unique.*
- *Dans la situation du problème $(PLMO)$, l'ensemble Z est convexe ; la résolution du problème paramétrique permet donc de déterminer l'ensemble de toutes les solutions efficaces. Cependant, la stucture discrète des problèmes d'OCMO rend ce résultat invalide. Une conséquence essentielle de ce fait, est qu'il n'est plus suffisant d'agréger les objectifs par une somme pondérée lorsqu'on tente de déterminer l'ensemble de toutes les solutions efficaces (ou points non dominés dans l'espace des objectifs). Ainsi, il existe des solutions efficaces qui ne sont optimales pour aucune somme pondérée des objectifs. Ceci reste vrai dans les cas où la matrice des contraintes est totalement unimodulaire contrairement à la proposition dans [47] (voir [86] par exemple).*

Ces considérations suggèrent l'introduction de quelques nouvelles définitions de sous ensembles de solutions efficaces et points non dominés.

Définition 1.12. Soient E l'ensemble de solutions efficaces dans l'espace des solutions, et Z_E l'ensemble des points non dominés dans l'espace des objectifs.

1. Soit $x \in E$. Si il existe $\lambda^0 \in \Lambda$ tel que $x \in E$ soit une solution optimale du problème paramétrique $\min_{x \in S} \sum_{k=1}^{m} \lambda_k^0 f^k(x)$, alors x est appelée *solution efficace supportée* et $z = F(x)$ est appelée *point non dominé supporté*. Les ensembles de toutes les solutions efficaces supportées et tous les points non dominés supportés sont notés SE et SZ_E, respectivement. Sinon x et z sont dits "non supportés". Les ensembles de solutions efficaces et points non dominés non supportés sont noté $NS(E)$ et $NS(Z_E)$.
2. x_1 et x_2 sont dites équivalentes si $F(x_1) = F(x_2)$.
   - Un ensemble *complet de solutions efficaces* est tout sous ensemble $S' \subset S$ tel que $F(S') = Z_E$.
   - Un ensemble *complet minimal* est un ensemble complet ne possédant pas de solutions équivalentes.
   - E est appelé aussi *ensemble complet maximal*

3. Si x est une solution efficace supportée et $z = F(x)$ un point extrème de l'enveloppe convexe $convZ$ alors x est appelée *solution efficace supportée extrème* et z est un *point non dominé extrème*

Lorsque nous abordons la résolution des problèmes d'OCMO, il est trivial voir même important, de noter que E et par suite Z_E sont des ensembles finis. Par conséquent, quelle que soit la classe des problèmes d'OCMO considérée, il existe toujours des solutions efficaces car Z_E n'est jamais vide. Une autre difficulté fondamentale dans l'OCMO est le nombre de solutions non dominées. Ce nombre est peut être très élevé et croît exponentiellement avec la taille du problème, interdisant à toute méthode "efficace" à déterminer toutes les solutions efficaces. En effet, pour beaucoup de problèmes il est en réalité possible de créer des instances où toutes les solutions réalisables sont non dominées. Comme par exemple : considérons un problème bi-objectif incluant des instances tels que $f_2(x) = k - f_1(x)$ où k est une constance. Pour ces instances toutes les solutions réalisables seront non-dominées. Ces résultats sont également connus pour un certain nombre de problèmes d'OCMO tels que les problèmes : du plus court chemin, d'affectation, du voyageur de commerce, etc...Par conséquent tels problèmes sont dits *intractable*. Les premiers articles se référant à l'OCMO ne se sont interessés qu'à la détermination de sous ensembles de solutions supportées. Néanmoins, l'ensemble $NS(E)$ des solutions non supportées est important et contribue essentiellement à la difficulté de ces problèmes. Les résultats numériques [93] montrent que le nombre des solutions non supportées est habituellement beaucoup plus élévé que celui des solutions supportées. La taille de l'ensemble de solutions efficaces supportées peut être exponentielle (voir [72], [38] ,[26]). Cependant, les résultats numériques disponibles sur le problème du sac-à-dos montrent que le nombre de solutions efficaces supportées croît linéairement avec la taille du problème mais que le nombre de solutions efficaces non supportées croît suivant une fonction exponentielle [93].

1.4 Approches de résolution des problèmes d'OCMO

Dans le contexte de la programmation multi-objectif, il est habituel de distinguer les méthodes suivant le rôle du décideur dans le processus de résolution. Souvent, l'information fournie par le décideur concerne ses préférences. Dans le mode "a priori", toutes les préférences sont connues au début du processus d'Aide à la Décision. Les techniques utilisées, calculent les solutions sur la base des paramètres prédéfinies. Le meilleur exemple est donné par les méthodes du "goal programming". Dans le mode "a posteriori", l'ensemble de toutes les solutions efficaces est généré pour le problème considéré. A la fin, cet ensemble

est analysé suivant les préférences du décideur. Plusieurs méthodes approximatives sont conçues suivant ce mode de résolution. Dans le mode "intéractif", les préférences sont introduites par le décideur durant le processus de résolution. Les méthodes nécéssitent des séries d'étapes de calculs alternées avec des étapes de dialogue. Elles peuvent être vues comme une détermination interactive d'un compromis satisfaisant pour le décideur et éxigent un haut niveau de participation de la part de ce dernier. Les problèmes pratiques sont souvent résolus suivant le mode "intéractif". Le mode de résolution approprié est choisi en considérant la situation du processus décisionnel. Et la méthode impliquée dans le processus peut être exacte ou approximative. Parmi toutes ces méthodes, il faut distinguer deux catégories : les approches non Pareto et les approches Pareto. Les approches non Pareto ne traitent pas le problème comme un véritable problème multi-objectif. Elles cherchent à ramener le problème initial à un ou plusieurs problèmes mono-objectifs. Par opposition aux approches non Pareto, les approches Pareto ne transforment pas les objectifs du problème.

1.5 Quelques méthodes exactes de résolution des problèmes d'OCMO

On remarque tout abord qu'il y a très peu de travaux sur les méthodes exactes dans le contexte de la résolution des problèmes d'optimisation multi-objectifs NP-difficiles, sans doute, ceci est du à la grande difficulté de ce type de problème. Les références ([85], [86], [93] et [25]) présentent la plupart des méthodes exactes existantes. Dans ce qui suit, nous décrirons les principales approches non Pareto exactes et commentons leurs avantages et leurs inconvénients.

1.5.1 Méthodes d'agrégation des objectifs

La démarche générale dans cette méthode consiste à agréger les objectifs en une fonction unique qu'il s'agira ensuite d'optimiser. Il existe plusieurs formes particulières de la fonction d'agrégation construite à l'aide des paramètres de préférence connus ou déterminés tels que :

- **Le poids** λ_i, qui reflètent l'importance relative de chaque objectif.
- **Le point de référence**, qui reflète les valeurs souhaitables sur chaque objectif.
- **Le point de réservation**, qui reflète les valeurs non souhaitables sur chaque objectif.

En tenant compte d'un jeu de poids de préférence -donné ou évalué- il est donc possible de construire une *fonction d'utilité*. Il s'agit d'une fonction : $U(z,\lambda) : Z \times \Lambda \rightarrow \Re$ Où :

- Z est l'espace des objectifs,
- $\Lambda \subset \Re^m$, est l'ensemble de poids.

L'objectif dans cette méthode est de trouver un point admissible le plus proche possible du point de référence (par exemple le point idéal) ou le plus éloigné du point de réservation (par exemple le point nadir). Les exemples les plus courants de fonctions d'utilité sont les suivants :

□ **Caractérisation à l'aide de poids**

- Fonctions linéaires :

$$U_1(z,\lambda) = \sum_{k=1}^{m} \lambda_k z^k \tag{1.2}$$

$$U_2(z,\lambda) = \sum_{k=1}^{m} \lambda_k |z^k - \widetilde{z}^k| \tag{1.3}$$

  avec $\sum_{k=1}^{m} \lambda_k = 1; \lambda_k > 0; k = 1..m$

- Norme L_p pondérée :

$$U_3(z,\lambda) = [\sum_{k=1}^{m} \lambda_k |z^k - \widetilde{z}^k|^p]^{\frac{1}{p}} \tag{1.4}$$

  avec $p \in Z_0^+$.

- Norme L_∞ de Tchebytcheff pondérée :

$$U_4(z,\lambda) = \max_{1 \leq k \leq m} \{\lambda_k |z^k - \widetilde{z}^k|\} \tag{1.5}$$

- Norme composite (Tchebytcheff pondérée augmentée) :

$$U_5(z,\lambda) = \max_{1 \leq k \leq m} \{\lambda_k |z^k - \widetilde{z}^k|\} + \rho \sum_{k=1}^{m} \lambda_k |z^k - \widetilde{z}^k| \tag{1.6}$$

  avec $0 < \rho$.

□ **Caractérisation à l'aide de points cibles**

- Niveaux d'aspiration :

$$U_6(z,\lambda) = [\sum_{k=1}^{m} \lambda_k |z^k - \widehat{z}^k|^p]^{\frac{1}{p}} \tag{1.7}$$

  avec $p \in Z_0^+$ et $\widehat{z} \in Z$.

– Niveaux de réservation :

$$U_7(z,\lambda) = [\sum_{k=1}^{m} \lambda_k (z^k - \widehat{z}^k)^p]^{\frac{1}{p}} \tag{1.8}$$

et des contraintes sur les objectifs $z^k \geq \widehat{z}^k$, pour tout $k = 1..m$.

Il est clair que la résolution du problème pour un vecteur de poids λ fixé ne permet de calculer que quelques solutions efficaces. Pour obtenir un ensemble contenant un grand nombre de solutions efficaces, il faut résoudre plusieurs fois le problème en changeant à chaque fois les valeurs de λ. Cette approche a l'avantage évident de pouvoir réutiliser tous les algorithmes classiques dédiés aux problèmes d'optimisation à un seul objectif. C'est souvent la première approche adoptée lorsqu'un chercheur se retrouve devant un nouveau problème multi-objectif. Cependant cette approche a aussi deux inconvénients importants. Le premier est dû au fait que pour avoir un ensemble de points bien répartis sur le front Pareto, les différents vecteurs λ doivent être choisis judicieusement. Il est donc nécessaire d'avoir une bonne connaissance du problème. Le deuxième inconvénient provient du fait que cette méthode ne permet pas, d'identifier la totalité du front Pareto lorsque celui-ci est non convexe. C'est particulièrement le cas pour les problèmes d'OCMO.

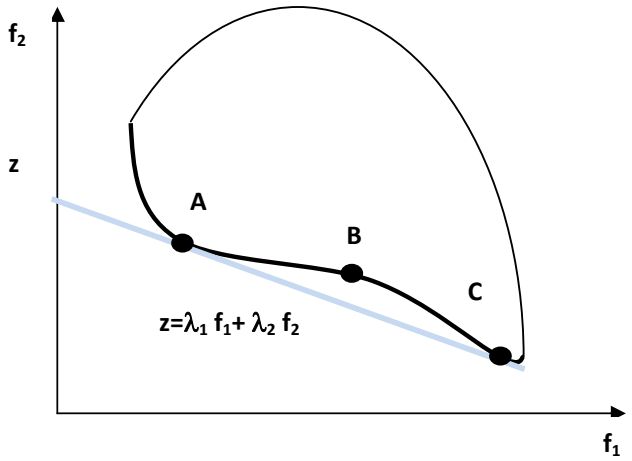


FIG. 1.3: Interprétation graphique des méthodes d'agrégation des objectifs

La figure 1.3 montre que pour un vecteur de poids $\lambda = (\lambda_1, \lambda_2)$ fixé, les deux points Pareto optimaux trouvés sont A et C. En faisant varier le vecteur λ, il est possible de trouver d'autres points Pareto optimaux. Seulement, tous ces points se trouveront sur les parties convexes de la surface de compromis. Il n'existe pas de valeur possible pour λ

permettant de trouver les points situés sur la partie concave de la surface de compromis, par exemple le point B.

1.5.2 Méthode de seuils de satisfaction ou ϵ-contrainte

La démarche générale dans cette méthode et de transformer un problème d'optimisation multi-objectif en un problème à un seul objectif, en convertissant les $m-1$ des m objectifs du problème en contraintes, ce qui aura pour effet de diminuer la zone réalisable, et en optimisant séparément l'objectif restant. Le problème peut être reformulé de la manière suivante :

$$(P_{\epsilon_2}) \begin{cases} min\ f^1(x) \\ s.c \\ f^k(x) \leq \epsilon_k \quad k = 2..m \\ x \in S \end{cases}$$

$\epsilon_1, \epsilon_2, ..., \epsilon_m$ sont des niveaux maxima pouvant être atteints par les objectifs $f^1, f^2, ..., f^m$ respectivement. Ils sont spécifiés par le décideur. D'autre part n'importe quel objectif peut être pris comme objectif principal f^1. Pour trouver un ensemble de solutions efficaces, la méthode ϵ-contrainte est appliquée plusieurs fois en faisant varier le vecteur ϵ. Cette méthode a l'avantage par rapport à la précédente de ne pas être limitée aux problèmes convexes. En effet, la figure 1.4 illustre, en dimension 2, le cas où un point de la partie non convexe, est trouvé. Mais possède le même inconvénient lié au fait qu'il faille lancer un grand nombre de fois le processus de résolution. Et qu'une bonne connaissance du problème *a priori* est requise.

1.5.3 Méthode du Goal programming

La première tentative de la méthode du Goal programming en programmation mathématique multi-objectif est due à Charnes et Cooper (1961). Cette méthode repose sur le schéma de base suivant :

- Fixer les valeurs que l'on désire atteindre sur chaque objectif (ce sont les buts),
- Associer des poids à ces buts,
- Définir des variations (positives et négatives) par rapport à ces buts,
- Effectuer une analyse de sensibilité.

Cette méthodologie a été d'abord développée dans le cadre de la programmation linéaire multi-objectif. Elle a été ensuite étendue à d'autres types de programmes multi-objectifs,

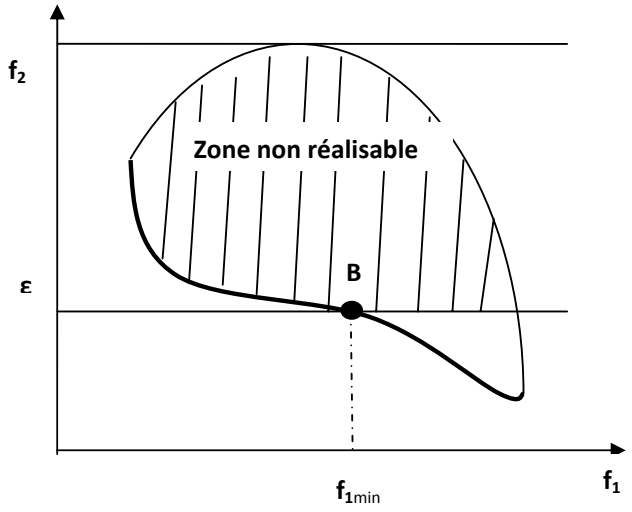


FIG. 1.4: Interprétation graphique de la méthode ϵ-contrainte

puis a été rendue interactive. Les buts peuvent être des valeurs spécifiques ou des rangs de telles valeurs et peuvent se rapporter à des propriétés métriques comme à des propriétés non métriques. Dans les deux cas, à chaque but est attribuée une fonction "objectif" qui pénalise les déviations du but souhaité. Ces fonctions sont, par conséquent, appelées "fonctions buts". Le vecteur des fonctions buts est soumis à une minimisation. Ce vecteur peut être de type scalaire, et peut renfermer des combinaisons des différentes fonctions but. De nombreuses applications du "Goal Programming" ont été mises en oeuvre dans divers domaines et en particulier dans celui de l'OCMO ([42], [51]).

(a) Modèle archimédien. Dans les modèles archimédiens les objectifs sont agrégés par une combinaison linéaire. Le vecteur des "fonctions but" est :

$$\min_{x \in S} \sum_{k=1}^{m} | \sum_{j=1}^{n} f_j^k x_j - g_k |. \tag{1.9}$$

Les buts g_k sont des constantes fixées. On peut remplacer la formulation (1.9) par :

$$(1.9a) \begin{cases} min \sum_{k=1}^{m} (d_k^+ + d_k^-) \\ s.c \\ \sum_{j=1}^{n} f_j^k x_j - d_k^+ + d_k^- = g_k \quad k = 1..m, \\ d_k^+, d_k^- \geq 0, \\ x \in S. \end{cases}$$

Pour montrer l'équivalence entre les formulations (1.9) et (1.9a), nous définissons :

$$(1.9b)\begin{cases} d_k^+ = \frac{1}{2}[|\sum_{j=1}^n f_j^k x_j - g_k| + (\sum_{j=1}^n f_j^k x_j - g_k)], \\ d_k^- = \frac{1}{2}[|\sum_{j=1}^n f_j^k x_j - g_k| - (\sum_{j=1}^n f_j^k x_j - g_k)], \end{cases}$$

En additionnant et en soustrayant les expressions entre parenthèses à (1.9) et en collectant les termes, on obtient la forme (1.9a) avec la non-négativité de chacune des paires (d_k^+, d_k^-). En soustrayant d_k^+ de d_k^-, on obtient les relations de contraintes :

$$\sum_{j=1}^{n} f_j^k x_j - d_k^+ + d_k^- = g_k, \quad k = 1..m. \tag{1.10}$$

Comme on peut le constater, $d_k^+.d_k^- = 0$ pour tout $k = 1..m$. En fait, ces conditions, qui ne sont d'ailleurs non linéaire, n'ont pas besoin d'être maintenues dans (1.9a) du moment où seulement une équivalence à l'optimum est souhaitée. La paire (d_k^+, d_k^-) peut être interprétée comme une mesure de distance (ou d'écart) relative au k^{eme} but. Ainsi, l'objectif dans (1.9a) et d'où dans (1.9) est d'effectuer le choix des variables de décision qui minimisent la somme des écarts. En toute évidence, ont peut pondérer les écarts (d_k^+ et d_k^-), différemment comme suit :

$$(1.11)\begin{cases} min \sum_{k=1}^m (\lambda_k^+ d_k^+ + \lambda_k^- d_k^-) \\ s.c \\ \sum_{j=1}^n f_j^k x_j - d_k^+ + d_k^- = g_k \quad k = 1..m, \\ d_k^+, d_k^- \geq 0, \\ x \in S. \end{cases}$$

où les λ_k^+, λ_k^- sont des constantes positives représentants les poids relatifs, respectivement, aux déviations positives et négatives pour chaque but g_k.

Analyse de sensibilité

En pratique, on fait joindre des analyses de sensibilité à la résolution du modèle (1.11).

- Variation des fonctions objectifs en fonction de λ_k^+ et λ_k^- : Elle consiste à faire varier les valeurs λ_k^+ et λ_k^- et à enregistrer les variations des fonctions objectifs.

(b) Modèle lexmin. Dans le modèle lexmin les buts sont rangés suivant un ordre de priorité.

Définition 1.13. Le minimum lexicographique. Un vecteur $u = (u_1, u_2, ..., u_n)$ est dit *lexicographiquement plus petit que* $v = (v_1, v_2, ..., v_n)$ si et seulement si il existe un indice $j \in \{1, 2, ..., n\}$ pour lequel $u_j < v_j$ et $u_i = v_i$ pour tout $i \in \{1, 2, ..., j-1\}$

Remarque 1.2. *Le minimum lexicographique d'un ensemble S de vecteurs, est un vecteur u^* dans S qui est lexicographiquement plus petit que tous les vecteurs dans S différent de u^*. On note $u^* = lexmin(S)$.*

Un modèle lexmin est généralement formulé comme suit : lexmin$(u = (u_1, u_2, ..., u_n))$, avec $u \in S$.

Résolution d'un problème lexmin. Dans le cas des variables linéaires, les problèmes lexmin peuvent être résolus en résolvant n programmes linéaires de la manière suivante : Soit j, $1 \leq j \leq n$, déterminer successivement :
$u_1^* = \min\{u_1/u \in S\}$,
$u_2^* = \min\{u_2/u \in S, u_1 = u_1^*\}$
$\vdots$
$u_j^* = \min\{u_j/u \in S, u_1 = u_1^*, u_2 = u_2^*, u_{j-1} = u_{j-1}^*, \}$

La séquence s'arrête après j étapes si j est le plus petit indice pour lequel le programme linéaire correspondant possède une solution unique.

1.5.4 Méthode de ranking

Une approche très populaire pour les problèmes bi-objectif est l'utilisation des méthodes de "ranking". Le point idéal $\overline{\pi} = (\overline{\pi}_1, \overline{\pi}_2)$ et le point nadir $\underline{\pi} = (\underline{\pi}_1, \underline{\pi}_2)$ définissent, respectivement, une borne inférieure et une borne supérieure sur les valeurs des objectifs des solutions efficaces.

En commençant par la solution ayant $z^1(x) = \overline{\pi}_1$, et en recherchant la seconde meilleure, la troisième meilleure, ..., la K-meilleure solutions suivant le premier objectif jusqu'à ce que la borne $\underline{\pi}_1$ soit atteinte, l'ensenble des solutions efficaces peut être déterminé.

Cette méthode a été appliquée pour le problème du plus court chemin ([54], [55]), le problème de transport ([55]) et le problème d'affectation [63]. Notons que le calcul du point nadir $\underline{\pi}$ dans le cas bi-objectif est essentiellement la solution de deux problèmes d'optimisation lexicographiques. La généralisation de cette méthode pour plus de deux objectifs n'est pas possible sans la connaissance du point nadir qui est difficile à déterminer lorsque $m > 2$ (voir [49]). Notons également que la généralisation du point nadir (énoncée sans preuve dans [54]) ne fournit pas nécéssairement une borne superieure sur

les valeurs des objectifs des solutions efficaces. En effet, même la considération de l'optimisation lexicographique suivant toutes les permutations des objectifs, ne guarantit pas la détermination des bornes superieures sur les valeurs des objectifs des solutions efficaces.

1.5.5 Méthode des deux phases

La méthode des deux phases est un schéma général pour la résolution exacte des problèmes d'optimisation combinatoire bi-objectif et la détermination de l'ensemble des solutions efficaces dans ce cas. Son nom remonte aux travaux d'Ulungu [85] et [86]. Dans la première phase, l'ensemble des solutions efficaces supportées est déterminé en utilisant les techniques d'agrégation des objectifs et en exploitant les méthodes efficaces disponibles pour le cas mono-objectif. La deuxième phase consiste à calculer l'ensemble des solutions efficaces non supportées par des méthodes spécifiques au problème considéré (en utilisant des bornes, des coûts réduits, etc). En effet, aucune caractérisation générale et théorique pour le calcul des solutions non supportées n'existe à ce jour.

1.6 Méthodes approchées de résolution des problèmes d'OCMO

Ces trois dernières décennies ont connu des développements et des améliorations fulgurents des méthodes de résolution approximatives, appelées habituellement "heuristiques et métaheuristiques". Une heuristique est définie par [66], comme une technique qui détermine de bonnes solutions (i.e., proches de l'optimum) en un temps de calcul raisonnable sans pour autant guarantir la réalisabilité ou l'optimalité ou même dans de nombreux cas connaître sa proximité de la solution optimale. La plupart d'entre elles sont conçues pour des problèmes spécifiques et ne peuvent être utilisées pour résoudre d'autres types de problèmes. Au contraire, les métaheuristiques sont des techniques puissantes qui s'adaptent à différents types de problèmes combinatoires ou même continus. Leurs succès est du à la capacité de telles techniques de "résoudre en pratique quelques problèmes combinatoires durs". Dans ce paragraphe, nous nous intéressons à la résolution approchée de problèmes d'OCMO NP-difficiles, notamment par des métaheuristiques. Une liste de 1380 références sur la théorie et l'application des métaheuristiques est présentée dans [59] et une présentation plus détaillée des métaheuristiques sont donnée dans ([66],[40], [18], [64],[24]).

Les caractéristiques principales qui semblent différencier les diverses métaheurisques pour l'optimisation multi-objectif sont : d'une part, le principe du mécanisme de recherche pour développer des solutions (diversification au moyen de re-combinaisons ou de recherches locales) et d'autre part, les méthodes utilisées pour évaluer et comparer les solutions (agrégation ou relation de dominance). Un examen sur l'application de ces métaheurisques pour approximer la frontière des solutions efficaces révèle que les "algorithmes evolutionnaires" tels que :"vector evaluated genetic algorithm" (VEGA)[67], "multi objective genetic algorithm" (MOGA)[29], "nondominated sorting genetic algorithm" (NSGA) [75], "strength Pareto evolutionary algorithm"(SPEA) [96], et "memetic Pareto archived evolution strategy algorithm" (M-PAES) [19], ont reçu récemment une attention considérable.

Ils semblent être couronnés de succès mais n'ont pas encore été suffisamment évaluer dans des problèmes combinatoires difficiles. En effet, un grand nombre de tests et comparaisons a été effectué essentiellement pour des problèmes continus. Cependant, peu d'études a été réalisé en optimisation combinatoire [7] et [10]. Les opérateurs génétiques sont les principaux mécanismes pour le développement des solutions et dans la plupart de ces approches, la relation de dominance est utilisée dans le processus de recherche.

Il a été noté, également, que les métaheuristiques basées sur "la recherche locale" telles que le recuit simulé [88] et la recherche tabou [30] sont largement utilisées dans les problèmes difficiles d'optimisation combinatoire multi-objectif [9], [8], [13], [14] et [16].

Quelques études ont été recensées sur les problèmes continus. La recherche du voisinage est le principal mécanisme pour développer des solutions et dans la plupart de ces approches, une méthode d'agrégation est utilisée, même lorsqu'on souhaite déterminer un ensemble de solutions.

Dans leur démarche unificatrice de la programmation à mémoire adaptative [83] , E. Taillard et al. remarquent que les métaheuristiques reposent finalement sur un ensemble commun de concepts peu nombreux : la mémoire, qui sauvegarde l'information receuillie par l'algorithme, l'intensification, qui tente d'améliorer la pertinence des informations disponibles, et la diversification, qui vise à accroître la quantité de ces informations, en explorant de nouvelles régions de l'espace de recherche. Les associations multiples de ces concepts peuvent conduire à une grande variété de métaheuristiques : la filiation à une

métaphore donnée n'est alors plus nécessairement explicite.

1.6.1 Le recuit simulé

Le recuit simulé s'inspire du processus de recuit physique [48]. 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. Le recuit simulé a acquis son succès notamment grâce à des résultats pratiques obtenus sur de nombreux problèmes dans des domaines variés. Des exemples de ces applications sont présentés dans ([1], [92], [46]). Une présentation détaillée de cette méthode est donnée dans le chapitre 3.

Adaptation de la méthode du recuit simulé au contexte multi-objectif

La méthode du recuit simulé a été adaptée de différentes manières au contexte multi-objectif, pour déterminer un sous ensemble de solutions potentiellement non dominées. Son utilisation comme une technique de résolution des problèmes d'optimisation multi-objectif a été analysée pour la première fois dans les travaux de Serafini [70], [71](SMOSA) (Serafini Multi Objective Simulated Annealing). L'approche utilisée consiste à combiner tous les objectifs du problème d'OCMO suivant :

$$(POCMO)\begin{cases} \text{“}min''\ F(x) = (f^1(x), f^2(x), ..., f^m(x)) \\ s.c \\ x \in S \end{cases}$$

dans une fonction mono-objectif en utilisant une fonction d'utilité U agrégeant les objectifs :

$$U(x, \lambda) = \sum_{k=1}^{m} \lambda_k f^k(x); \quad m \geq 2 \tag{1.11}$$

Cette fonction (1.11) est alors utilisée comme une “énergie” à minimiser dans un recuit simulé mono-objectif classique [48].

Considérons une solution courante x_r et soit $y \in V(x_r)$, une solution générée aléatoirement dans le voisinage $V(x_r)$ de x_r. En calculant $\Delta U = U(y, \lambda) - U(x_r, \lambda)$, la stratégie de sélection de la nouvelle solution courante consiste en les décisions suivantes :

a) si $\Delta U < 0$ alors $x_{r+1} \leftarrow y$.

b) si $\Delta U \geq 0$ alors

   1. $x_{r+1} \leftarrow y$ avec une probabilité p

2. $x_{r+1} \leftarrow x_r$ avec une probabilité $1 - p$.

A chaque itération, la direction de recherche λ est modifiée aléatoirement. Malheureusement, il est connu que les algorithmes basés sur des méthodes d'agrégation des fonctions objectifs ne sont pas capables de générer des solutions non supportées c'est à dire les solutions non dominées situées dans les portions concaves de la frontière Pareto. Une autre adaptation de la méthode du recuit au contexte multi-objectif fut proposée par Ulungu [88] MOSA (Multi Objective Simulated Annealing). Dans son approche, Ulungu utilise directement la notion de dominance pour la comparaison des solutions générées. Toutefois, comme dans le cas de SMOSA (Serafini Multi Objective Simulated Annealing), il ne lui a pas été possible d'éviter de faire appel, dans la règle de Métropolis, à une fonction d'agrégation des écarts et donc à une direction de recherche définie à priori. Pour palier à cet inconvénient, et diversifier la recherche aléatoire dans d'autres directions, un processus de recuit simulé indépendant est défini suivant un certain nombre "J" de directions de recherche prédéfinies et diversifiées. Plusieurs listes de solutions potentiellement non dominées $\widehat{E}_1, \widehat{E}_2, \ldots$ sont alors générées suivant les directions de recherche $\lambda^1, \lambda^2, \ldots$ et fusionnées pour fournir $\widehat{E}$. Une version interactive de MOSA a été utilisée pour résoudre un problème industriel [90].

D'autres adaptations de la méthode du recuit simulé au contexte multi-objectif

D'autres adaptations du recuit simulé au contexte multi-objectif ont été proposées :

- Une alternative équivalente [27] à la méthode MOSA de Sérafini utilise la somme des $\log c^k(x)$ et d'autres ([58], [91]) ont investi le champs non linéaire et stochastique de la fonction d'utilité.
- Czyzak, Hapke et Jaszkiewicz PSA [21] ont combiné le recuit simulé mono-objectif et l'algorithme génétique pour résoudre le problème du plus court chemin multi-objectif.
- Suppapitnarm, Seffen, Parcs et Clarkson [81] ont proposé une approche de base différente pour le recuit simulé utilisé en multi-objectif.
- Hapke, Jaszkiewicz et Slowinski [43] ont étudié l'application de PSA (Pareto Simulated Annealing) pour l'optimisation combinatoire multi-objectif floue.
- Smith et al.[74] proposèrent un MOSA différent, en utilisant la dominance relative d'une solution comme un système d'énergie à optimiser et en éliminant le problème associé à la fonction d'agrégation (fonction d'utilité).
- Suman [77], [78], [79] a formulé deux algorithmes multi-objectifs basés sur le recuit simulé, à savoir le WMOSA et le PDMOSA, pour réduire au minimum le nombre

de paramètres. PDMOSA utilise uniquement une température de recuit et n'utilise pas de direction de recherche pour le calcul de la probabilité d'acceptation dans la règle de Métropolis et WMOSA utilise une direction de recherche prédéfinie et seulement une température de recuit pour le calcul de la probabilité d'acceptation dans la règle de Métropolis.

– Suman [80] a proposé un auto critère d'arrêt PDMOSA qui a été comparé avec le recuit simulé multi-objectif de Suppapitnarm.

1.6.2 La recherche Tabou

La recherche Tabou est introduite par Glover [32]. Cette méthode est amplement présentée dans [33]. Son fonctionnement est plus proche de la méthode de descente (amélioration itérative) que du recuit simulé. La recherche Tabou examine un échantillonnage de configurations de $V(x)$ et retient la meilleure x_0. Cependant, cette stratégie peut entraîner des cycles, par exemple le cycle de longueur 2 : $x \rightarrow x_0 \rightarrow x \rightarrow x_0$ .... Pour empêcher ce type de cycle d'apparaître, on mémorise les k dernières configurations visitées dans une mémoire à court terme et on interdit de retenir toute configuration qui en fait déjà partie. Cette mémoire, appelée la liste Tabou, est une des composantes essentielles de la méthode. En effet, elle permet d'éviter tous les cycles de longueur inférieure à k, k étant déterminée en fonction du problème. La mémorisation de configurations entières serait trop coûteuse en temps de calcul et en place mémoire. Il est préférable de mémoriser des caractéristiques des configurations au lieu de configurations entières. Plus précisément, il est possible de mémoriser uniquement un attribut de la configuration. Il en résulte que toutes les configurations possédant cet attribut, y compris celles qui n'ont pas encore été rencontrées, deviennent elles aussi Tabou. Pour palier à ce problème, un mécanisme particulier, appelé l'aspiration, est mis en place. Ce mécanisme consiste à révoquer le statut Tabou d'une configuration à certains moments de la recherche. La fonction d'aspiration la plus simple consiste à enlever le statut Tabou d'une configuration si celle-ci est meilleure que la meilleure configuration trouvée jusqu'alors. Il existe aussi d'autres techniques permettant d'améliorer les performances de la méthode Tabou, en particulier, l'intensification et la diversification. L'intensification permet de se focaliser sur certaines zones de l'espace de recherche en apprenant des propriétés favorables, par exemple les propriétés communes souvent rencontrées dans les meilleurs configurations visitées. La diversification a un objectif inverse de l'intensification. En effet, elle cherche à diriger la recherche vers des zones inexplorées, en modifiant par exemple la fonction d'évaluation.

L'intensification et la diversification jouent donc des rôles complémentaires. 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 : coloriage de graphe, sac à dos, sac à dos multidimensionnel . . . . La majorité des développements apportés ainsi qu'un large éventail d'applications de la recherche Tabou, se retrouvent dans [33].

1.6.3 Les algorithmes évolutionnaires

La deuxième classe de métaheuristiques est celle des algorithmes évolutionnaires [11]. On peut distinguer trois grandes classes d'algorithmes évolutionnaires : les algorithmes génétiques ([41], [36]), les stratégies d'évolution [68] et la programmation évolutive [28]. Ces méthodes se différencient par leur manière de représenter l'information et par leur façon de faire évoluer la population d'une génération à l'autre. Un algorithme évolutionnaire est typiquement composé de trois éléments fondamentaux :

- une population constituée de plusieurs individus représentant des solutions potentielles (configurations) du problème donné,
- un mécanisme d'évaluation des individus permettant de mesurer l'adaptation de l'individu à son environnement,
- un mécanisme d'évolution de la population permettant, grâce à des opérateurs prédéfinis, d'éliminer certains individus et d'en créer de nouveaux.

1.7 Méthodes hybrides de résolution des problèmes d'OCMO

Une autre façon d'améliorer les performances d'un algorithme ou de combler certaines de ses lacunes consiste à le combiner avec une autre méthode [82]. Ce principe général, appelé hybridation, peut s'appliquer pour un grand nombre de méthodes. Une multitude d'algorithmes hybrides ont fait leur apparition ces dernières années. Nous pouvons ainsi citer pour le cas des problèmes multiobjectifs : la méthode MOTS ([39],[30]) combinant une population et une recherche Tabou, la méthode PSA [22] combinant un algorithme génétique et le recuit simulé, la méthode M-PAES [19] intégrant un schéma généralisant l'implémentation d'un grand nombre d'algorithmes hybrides pour l'optimisation des problèmes multiobjectifs. Un cas particulier de l'hybridation entre deux méthodes consiste à combiner une méthode de recherche locale et une méthode exacte : la méthode du recuit simulé hybride [37] appliquée au problème "des enchères combinatoires" (à un seul

objectif) combine la métaheuristique du récuit simulé avec une méthode exacte du type "branch and bound". Suivant la manière dont elle effectue cette combinaison, la méthode hybride considérée donnera lieu soit à une méthode exacte soit à une méthode approchée.

1.8 Evaluation des performances des métaheuristiques

L'évaluation de la performance d'une métaheuristique à partir d'une représentation graphique est souvent imprécise et de plus est limitée à deux /trois objectifs. Au delà de trois objectifs, il apparaît nécessaire de définir des mesures quantitatives pour évaluer les performances des métaheuristiques développées pour l'optimisation multi-objectif. Peu de travaux présentés dans la littérature effectuent une évaluation quantitative des méthodes proposées. Cela vient du fait que la plus part de ces dernières ont été proposées pour des problèmes bi-objectif. Nous avons recensé dans la littérature un certain nombre de mesures de performance selon qu'elles soient basées sur l'approximation par rapport à la frontière efficace exacte ou quelles soient basées sur les préférences du décideur.

1.8.1 Mesures de performance basées sur l'approximation

Les différentes mesure suivantes permettent d'évaluer la qualité d'une approximation par rapport à la frontière Pareto exacte.

- Proportion de solutions efficaces générées par $\widehat{E}$ (E : ensemble de solutions efficaces) :

$$M_1 = \frac{|\widehat{E} \cap E|}{|E|}, \tag{1.12}$$

  L'inconvient majeur de cette mesure est qu'elle ne permet pas de comparer deux approximations, qui chacune ne contient pas de solutions efficaces.
- La distance moyenne entre les ensembles $\widehat{E}$ et E :

$$D_1(\widehat{E}, E) = \frac{\sum_{x \in E} d(\widehat{E}, x)}{|E|}, \tag{1.13}$$

  où $d(\widehat{E}, x) = \min_{\{x' \in \widehat{E}\}} d(x', x)$ et : $d(x', x) = \frac{1}{2} \sum_{k=1}^{2} |f^k(x') - f^k(x)|$
- La plus mauvaise distance entre $\widehat{E}$ and E :

$$D_2(\widehat{E}, E) = \max_{x \in E} d(\widehat{E}, x), \tag{1.14}$$

– La mesure d'uniformité de l'ensemble $\widehat{E}$:

$$\frac{D_2(\widehat{E}, E)}{D_1(\widehat{E}, E)}. \tag{1.15}$$

Cette mesure n'est pas significative lorsque l'approximation $\widehat{E}$, dans l'espace des objectifs, est concentrée sur une portion de la frontière efficace.

– Le temps d'execution représenté par CPU(t). Ce dernier depend de la complexité des opérateurs (structure de voisinage, recombinaison), de la machine, du language de programmation et du style de la programmation.

Toutes ces mesures requièrent la connaissance de la frontière Pareto exacte ou à défaut une très bonne approximation de cette dernière.

1.8.2 Mesures de performance basées sur des préférences

Pour la plupart des problèmes d'OCMO, il est difficile voir même impossible de déterminer la frontière Pareto exacte. Dans ce cas la qualité d'une approximation est mesurée par rapport à une autre approximation sur la base des préférences du décideur. Une famille de relations de comparaison (qualitative) ont été introduites par Hansen et Jaskiewicz en 1998 :

– Dominance faible : une approximation A *domine faiblement* une approximation B, si

$$A \neq B \text{ et } ND(A \cup B) = A,$$

où $ND(A \cup B)$ dénote l'ensemble des solutions non dominées dans $A \neq B$. Cette relation signifie que pour chaque solution $x_B \in B$ il existe une solution $x_A \in A$ qui est égale ou domine x_B, et au moins une solution de A n'est pas contenue dans B (figure 1.5).

– Dominance forte : une approximation A *domine fortement* une approximation B, si

$$ND(A \cup B) = A \text{ et } (B - ND(A \cup B) \neq \emptyset).$$

Cette relation signifie que pour chaque solution $x_B \in B$ il existe une solution $x_A \in A$ qui est égale ou domine x_B, et au moins une solution $x_B \in B$ est dominée par une solution $x_A \in A$ (figure 1.6).

– Dominance total : une approximation A *domine totalement* une approximation B, si

$$ND(A \cup B) = A \text{ et } (B \cap ND(A \cup B) = \emptyset).$$

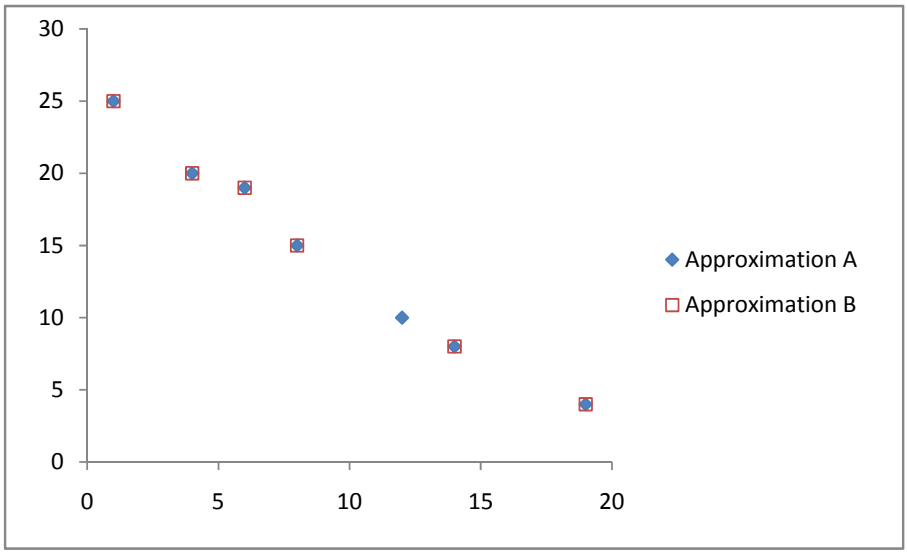


FIG. 1.5: Une approximation A domine faiblement une approximation B

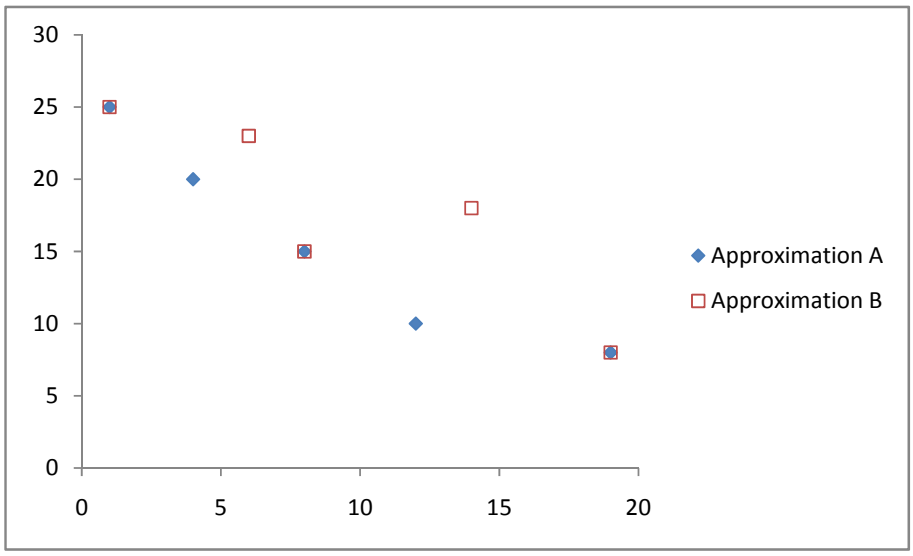


FIG. 1.6: Une approximation A domine fortement une approximation B

Cette relation signifie que pour chaque solution $x_B \in B$ est dominée par une solution $x_A \in A$ (figure 1.7).

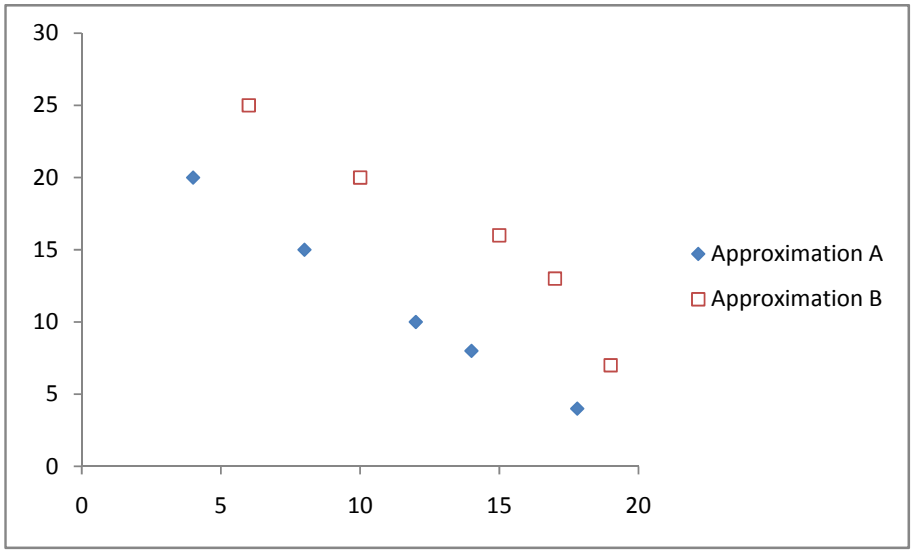


FIG. 1.7: Une approximation A domine totalement une approximation B

Les relations faible, forte et totale de dominance définissent un ordre partiel dans l'ensemble des approximations. La figure 1.8 montre un exemple d'approximations qui sont incomparables.

1.9 Conclusion

Une synthèse de différents travaux de recherche effectués les 30 dernières années dans le domaine de l'optimisation multi objectif, est présentée dans l'ouvrage de Ehrgott qui à notre connaissance est la plus récente référence dans ce domaine. Dans cette présente thèse, nous nous consacrons à l'étude du problème d'affectation multi-objectif, qui est un des problèmes classiques de l'optimisation combinatoire. Quoi que dans sa version mono-objectif ce problème est polynomial, il devient cependant NP-complet dans sa version multi-objectif.

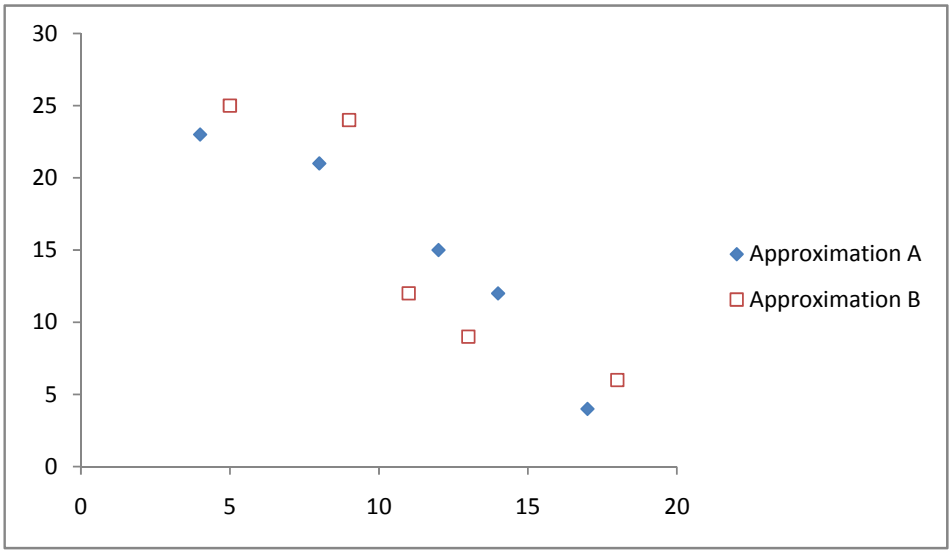


FIG. 1.8: Approximations incomparables

Chapitre 2

Une méthode "Branch and bound" pour la résolution du PAMO

2.1 Introduction

Toutes les approches exactes développées dans la littérature, pour la résolution du problème d'affectation bi-objectif, sont basées sur la méthode des deux phases qui est une technique générale pour la résolution des problèmes d'OCMO (voir [87]). Cependant, à notre connaissance, aucune des méthodes existantes ne permet de résoudre le problème au delà de deux objectifs [63]. Dans ce chapitre, nous proposons une adaptation de la méthode branch and bound au contexte multi-objectif (MOBB) [2], pour la résolution du problème d'affectation multi-objectif avec trois objectifs et plus.

2.2 Présentation du Problème d'Affectation Multi-Objectif (PAMO)

Le problème d'affectation peut être énoncé comme suit : n ouvriers doivent effectuer n tâches. Chaque ouvrier doit être affecté à une tâche et une seule et toutes les tâches doivent être attribuées. Si l'ouvrier O_i est affecté à la tâche T_j, il en résulte un coût c_{ij}. Le problème d'affectation, possède de nombreux liens avec d'autres problèmes d'optimisation combinatoire. En effet, il peut être vu comme un sous problème du problème de transport, du problème de distribution et du problème de couplage parfait de poids maximum dans un graphe. Il constitue également, une relaxation du célèbre problème du "voyageur de commerce" et peut donc servir à calculer une fonction d'évaluation dans une procédure

"branch and bound" pour ce problème.

La formulation mathématique du problème d'affectation en présence de m fonctions à optimiser s'écrit :

$$(PAMO)\begin{cases} \text{“ min ”} Z^k(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^k x_{ij} & k = 1..m \\ \sum_{j=1}^{n} x_{ij} = 1 & i = 1..n \\ \sum_{i=1}^{n} x_{ij} = 1 & j = 1..n \\ x_{ij} \in \{0,1\} & i = 1..n,\ j = 1..n \end{cases}$$

où c_{ij}^k sont des entiers non négatifs et $x = (x_{11}, ..., x_{nn})$ est une matrice binaire de variables de décision,

$$x_{ij} = \begin{cases} 1 & \text{si et seulement si l'ouvrier } O_i \text{ est affecté à la tâche } T_j\,; \\ 0 & \text{sinon}\,; \end{cases}$$

Considérons l'ensemble

$$S = \{x_{ij} \in \{0,1\}; i,j = 1..n |\ \sum_{j=1}^{n} x_{ij} = 1,\ i = 1..n,\ \sum_{i=1}^{n} x_{ij} = 1,\ j = 1..n\}$$

de solutions réalisables, défini par l'ensemble des contraintes du problème d'affectation multi-objectif (PAMO). Lorsqu'un seul objectif est considéré, la totale unimodularité de la matrice des contraintes garantit qu'une solution optimale entière est trouvée par les méthodes de la programmation linéaire. Avec la méthode hongroise un algorithme très efficace est disponible.

La problématique dans le contexte multi-objectif consiste à déterminer, de manière exacte ou approchée, l'ensemble des solutions efficaces. Toutefois, la détermination de manière exacte de l'ensemble des solutions efficaces est un problème NP-complet [69]. En effet, Sérafini propose, en 1987, une définition de la NP-complétude pour les problèmes d'optimisation combinatoire multi-objectif et démontre également comment certains problèmes d'optimisation combinatoire mono-objectif "faciles" deviennent NP-complets dans le cas bi-objectif. C'est particulièrement le cas pour le problème d'affectation et le problème du plus court chemin.

2.3 Limitation des méthodes exactes existantes

La plupart des approches exactes développées dans la littérature, pour la résolution du problème d'affectation bi-objectif, sont basées sur la méthode des deux phases qui utilise les méthodes mono-objectifs et les propriétés de dualité du problème d'affectation.

Cette méthode a été introduite, en premier lieu, en 1993 par Ulungu [85] qui a montré que les solutions efficaces ne sont pas connectées par les pivots du simplexe comme celà était initiallement supposé par Bhatia Malhotra et Puri [15]. En effet, des solutions efficaces non supportées existent. Dans le cas bi-objectif [84], elles sont localisées à l'interieur du triangle défini par deux solutions supportées adjacentes dans l'espace des objectifs (voir la figure 2.1).

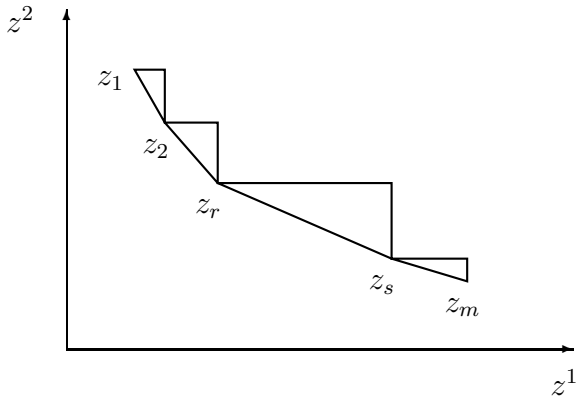


FIG. 2.1: Solutions efficaces supportées $(z_l, \ldots, z_m)$ et potentiellement non supportées $(\Delta z_r z_s)$.

Soit (PA_λ), le programme paramétrique suivant :

$$(AP_\lambda) \begin{cases} min\ z(x) = \sum_{k=1}^{m} \lambda_k Z^k(x) \\ x \in S \end{cases}$$

où $\lambda \in \Lambda$, avec $\Lambda = \{\lambda \in \Re^m \mid \sum_{k=1}^{m} \lambda_k = 1; \lambda_k > 0; k = 1..m\}$

Les solutions supportées sont identifiées dans la première phase, d'après le théorème de Geoffrion [35] qui peut être énoncé, pour le problème d'affectation multi-objectif, comme suit :

Théorème 2.1. *S'il existe $\lambda^0 \in \Lambda$ pour lequel $x^* \in S$ est une solution optimale du problème PA_{λ^0}, alors x^* est une solution efficace supportée du problème d'affectation multi-objectif $PAMO$.*

Plusieurs approches de la méthode des deux phases ont été proposées dans la littérature pour déterminer l'ensemble de toutes les solutions efficaces du problème d'affectation bi-objectif. Elles diffèrent principalement, dans leur manière de calculer, dans la deuxième phase, les solutions efficaces non supportées. Des résultats expérimentaux [23], basés sur la méthode de dichotomie et utilisant un solveur MIP, ont permis de détecter davantage de solutions efficaces. Przybylski et al.(2008) [63] ont calculé les solutions efficaces non supportées par une méthode de ranking et ont donné également une description complète de la méthode des deux phases pour le problème d'affectation bi-objectif. Cependant, à notre connaissance, aucune des méthodes existantes ne permet de résoudre le problème au delà de deux objectifs.

2.4 Principe des méthodes branch and bound

Les méthodes "branch and bound" font partie de la catégorie des méthodes exactes destinées à fournir toutes les solutions optimales de l'instance à traiter des problèmes d'optimisation combinatoire (OC), où les variables prennent des valeurs discrètes. Elles apparaissent pour la première fois sous cette appellation dans l'article de [53] consacré à la résolution exacte du problème du voyageur de commerce. Depuis, elles ont été appliquées à d'innombrables problèmes d'OC. Elles sont connues aussi sous d'autres noms : méthodes d'énumération implicite, procédures par séparation et évaluation progressive (SEP ou PSEP), procédures par séparation et évaluation séquentielle (SES ou SESP), diviser pour régner, algorithme A^* en intelligence artificielle ou encore sous des appellations moins habituelles telles que : énumération arborescente par séparation et élagage qui ne limite pas l'élagage au seul apport de l'évaluation. Ce sont des procédures qui examinent systématiquement toutes les combinaisons possibles des variables (discrètes), et utilisent intensivement certaines propriétés pour accélérer cet examen.

Les méthodes branch and bound consistent à construire une arborescence que nous notons $\mathfrak{F}$ dont chaque sommet S' représente une partie de l'ensemble S des solutions du problème. La racine R de $\mathfrak{F}$ est associée à l'ensemble S lui même $S(R) = S$. Le développement de $\mathfrak{F}$ depuis la racine se fait à l'aide de trois procédures essentielles :

- Procédure de séparation.
  - Définir la partition de l'ensemble fondamental.
  - Choisir le sous ensemble à traiter.
- Procédure d'évaluation.
  - Comment évaluer chaque sous ensemble.

– Procédure de stérilisation .
  – Quand un sous ensemble peut être stérilisé ?

Cependant, si le schéma général est tout à fait désigné pour trouver une solution optimale dans le cas mono-objectif, il n'offre, dans sa forme initiale, aucune perspective dans le contexte multi-objectif, où la préoccupation est tout à fait différente. Il ne s'agit plus de chercher une solution, optimale, mais d'en trouver tout un ensemble. La méthode que nous allons décrire dans la section suivante, est une adaptation au cas multi-objectif du schéma général des méthodes branch and bound.

2.4.1 Initialisation de la méthode branch and bound

En résolvant séparément le problème mono-objectif associé à chaque objectif, par la méthode hongroise, nous obtenons alors toutes les composantes du point idéal. Nous pouvons initialiser alors, l'ensemble des solutions efficaces, E, par l'ensemble des solutions optimales non-dominées. En effet, si pour un certain objectif, la solution optimale n'est pas unique ou encore si une solution est optimale pour plus d'un objectif alors on choisit celle qui n'est pas dominée suivant les autres objectifs.

2.4.2 Procédure de séparation

La procédure de séparation définit la façon de construire les sommets S' de $\mathfrak{F}$ et les ensembles $E(S')$ associés. Généralement, elle est issue d'un principe d'énumération implicite permettant de ne pas oublier de solutions. Par exemple, en testant les unes après les autres toutes les valeurs que peut prendre les variables du problème et qui sont en nombre fini et facilement déterminables, ou à interdire certaines valeurs. Ou encore, à partitionner l'ensemble des valeurs possibles d'une variable en sous ensembles. Cependant, la procédure de séparation peut résulter de considérations autres qu'une pure énumération exhaustive. Lorsque, par exemple, on est amené à changer la formulation du problème et à engendrer progressivement de nouveaux problèmes (ajouter progressivement de nouvelles contraintes). Soit S' le sommet de $\mathfrak{F}$ auquel on souhaite appliquer le principe de séparation, et soit $E(S')$ l'ensemble de solutions qui lui est associé. Séparer S', c'est créer un certain nombre $k(S')$ (dépendant à priori de S') de sous ensembles $S'_1, S'_2, \ldots, S'_k$ dont l'union est égale à $S' : \bigcup_{i=1}^{k} S'_k = S'$.

Pour le problème d'affectation multi-objectif, la procédure de séparation ne pose aucun problème et ne nécessite aucune adaptation. Une manière commode d'énumérer sans redondance l'ensemble des $n!$ affectations, est la décomposition par étapes. A la première

étape, la tâche T_1 est affecté à l'un des n ouvriers. A la seconde étape, la tâche T_2 est affecté à l'un des $n-1$ ouvriers restants et ainsi de suite. L'ordre selon lequel sont choisis les sous ensembles à traiter est fixé par le type de parcours adopté pour développer l'arborescence. Ici, on adopte un parcours hybride qui combine les deux parcours en largeur et en profondeur. On commence la construction de l'arborescence par un parcours en largeur. On crée tous les successeurs de la racine sans se soucier, dans un premier temps, si certains de ces successeurs seraient éliminés, ou pas, par la stérilisation. Chaque nœud du premier niveau serait examiné selon l'ordre de sa création, et sera entre autre évalué. Dans le cas où il n'est pas stérilisé, on crée en profondeur tous ces successeurs qui appartiendront au deuxième niveau. On reviendra ensuite, aux nœuds du premier niveau qui n'ont pas été encore examinés, et ainsi de suite.

2.4.3 Procédure d'évaluation

Une fonction d'évaluation d'un problème d'Optimisation Combinatoire Multi-Objectif (OCMO) est un vecteur à la fois rapide à calculer (en temps polynomial) et qui soit proche de la surface de compromis.

Définition 2.1. Etant donné un problème d'optimisation combinatoire multi objectif.

- On dit qu'on sait évaluer le sous-ensemble S' de S si on sait déterminer un vecteur $g(S')$ ($g(S') = (g^1(S'), g^2(S'), \ldots, g^m(S'))$) tel que : il n'existe aucune solution $x \in S'$ telle que $Z(x)$ domine $g(S')$. Autrement dit, il n'existe pas $x \in S'$ tel que $Z^k(x) \leq g^k(S')$ pour tout $k = 1..m$ et $Z^k(x) < g^k(S')$ pour au moins un objectif (pour un problème de minimisation).
- Si $S' = \emptyset$ alors la seule évaluation possible est $g(S') = +\infty$.
- Une évaluation $g(S')$ est dite exacte si pour au moins un objectif $k' \in \{1, \ldots, m\}$, on connaît $x^* \in S'$ tel que $g^{k'}(S') = Z^{k'}(x^*)$.

La définition d'une procédure d'évaluation dépend de la nature du problème à résoudre.

Considérons un problème d'affectation avec m objectifs. La k-ème composante, $g^k(S)$, du vecteur évaluation de l'ensemble S, $g(S)$, est égale à la somme des minimums des colonnes de la matrice C_k ($k = 1..m$). Chaque composante $g^k(S)$, ($k = 1..m$) ainsi définie constitue une borne inférieure de la valeur de la fonction objectif $Z^k(x)$ de toute affectation réalisable x appartenant à l'ensemble S de toutes les affectations possibles.

Lemme 2.2. *Toute évaluation $g(S')$ ($S' \subset S$), peut être ajustée et améliorée par les valeurs du point idéal dans le cas où sa k-ème composante est inférieure à la valeur optimale du k-ème objectif.*

En effet, on peut résoudre séparemment, avec une méthode exacte, le problème mono-objectif associé à chaque objectif.

2.4.4 Procédure de stérilisation

Dans le contexte d'optimisation mono-objectif, la stérilisation d'un ensemble S' (correspondant à un nœud de l'arborescence d'exploration) signifie que cet ensemble ne peut contenir une solution optimale et il est inutile de s'y intéresser. Ceci peut être dû au fait qu'on dispose d'une solution au moins aussi bonne que toute (éventuelle) solution contenue dans S'. Dans un problème d'optimisation multi-objectif, une adaptation de ce concept est nécessaire.

Définition 2.2. Un sous-ensemble S' de l'ensemble S des solutions d'un problème d'optimisation combinatoire multi-objectif est dit stérilisé si :

- $S' = \emptyset$;
- ou bien il existe $x^* \in S$ tel que x^* domine toute solution de S', autrement dit : $Z^k(x^*) \leq g^k(S')$ pour tout $k = 1..m$, avec au moins une inégalité stricte.

Ainsi, il sera inutile de développer davantage le parcours d'un tel ensemble S', puisqu'on est assuré qu'il ne pourra contenir une quelconque solution efficace.

2.4.5 Algorithme de la méthode

L'algorithme que nous allons décrire est une adaptation au contexte multi-objectif du schéma général des méthodes par séparation et évaluation. La procédure de séparation ne pose aucun problème et ne nécessite aucune adaptation. La nouveauté essentielle consiste en la manière de parcourir l'arborescence. Celle-ci combine les deux types de parcours (en largeur et en profondeur). De plus, la notion de stérilisation est adaptée.

Nous initialisons l'Algorithme 1 avec m solutions efficaces supportées (non dominées) : $x_1^*, x_2^*, \ldots, x_m^*$, où x_k^* est une solution optimale suivant le k-ème objectif, $k = 1..m$. Chacune de ces solutions peut être calculée par la méthode hongroise.

2.5 Déroulement de l'algorithme

Nous allons appliquer le schéma que nous venons de décrire à l'exemple numérique suivant :

Algorithme 1 Branch and bound multi-objectif

$E := \{x_1^*, x_2^*, \ldots, x_m^*\}$,
$\mathfrak{F} := \{S\}$,
$Z(E) = \{Z(x_1^*), Z(x_2^*), \ldots, Z(x_m^*)\}$;
répéter
  Choix
  si un ensemble $S' \in \mathfrak{F}$ "non évalué" existe, **alors**
    le choisir ;
  sinon
    choisir un ensemble $S' \in \mathfrak{F}$ "évalué".
  finsi
  si S' n'a pas été "évalué" **alors**
    Evaluation : associer à S' son "evaluation" $g(S')$.
  finsi
  si il existe une solution $\widehat{e} \in E$ telle que $g(S')$ est "dominée" par $Z(\widehat{e})$ **alors**
    $\mathfrak{F} = \mathfrak{F} \setminus \{S'\}$.
  sinon
    si l'évaluation de S' est exacte (pour au moins un objectif) **alors**
      si $Z(x^*(S'))$ "domine" $Z(\widehat{e})$ **alors**
        $E := E \cup \{x^*(S')\} \setminus \{\widehat{e}\}$ (mise à jour de E).
        si $Z(x^*(S'))$ n'est "dominée" par aucune solution de E **alors**
          $E := E \cup \{x^*(S')\}$.
        finsi
      finsi
    finsi
  finsi
  Séparation : on crée les sous-ensembles $S'_1, S'_2, \ldots, S'_k$ et on pose $\mathfrak{F} := \mathfrak{F} \cup \{S'_1, S'_2, \ldots, S'_k\} \setminus \{S'\}$.
jusqu'à $\mathfrak{F} = \emptyset$

Considérons le problème d'affectation tri-objectif, dont les matrices coût C_1, C_2 et C_3 sont données ci-dessous :

$$\begin{bmatrix} \underline{1} & \underline{2} & 19 & 10 \\ 18 & 19 & \underline{4} & 5 \\ 4 & 11 & 17 & 12 \\ 2 & 14 & 19 & \underline{4} \end{bmatrix} \quad \begin{bmatrix} 15 & \underline{7} & 6 & 18 \\ 15 & 7 & 6 & 12 \\ \underline{9} & 15 & 6 & 18 \\ 15 & 13 & \underline{5} & \underline{3} \end{bmatrix} \quad \begin{bmatrix} 18 & 17 & 13 & 9 \\ 11 & 17 & \underline{2} & 7 \\ 8 & \underline{1} & 11 & 13 \\ \underline{3} & 7 & 15 & \underline{4} \end{bmatrix}$$

Nous calculons par la méthode hongroise les solutions optimales suivant chaque objectif (voir la table 2.1)

Objectifs	Solutions optimales	vecteurs coût
C_1	(3, 1, 2, 4)	$(14^*, 25, 31)$
C_2	(3, 2, 1, 4)	$(46, 25^*, 42)$
C_2	(3, 1, 2, 4)	$(14^*, 25, 31)$
C_3	(4, 3, 2, 1)	$(27, 54, 15^*)$

TAB. 2.1: Solutions optimales des problèmes mono-objectifs

La solution $s = (3, 1, 2, 4)$ signifie que : l'ouvrier O_3 est affecté à la tâche T_1, l'ouvrier O_1 est affecté à la tâche T_2, l'ouvrier O_2 est affecté à la tâche T_3 et l'ouvrier O_4 est affecté à la tâche T_4.

Le vecteur $Z^* = (14^*, 25^*, 15^*) \in \Re^3$, est le point idéal.

Nous pouvons observer dans Table 2.1 que le second objectif possède deux solutions optimales distinctes et que la solution ayant le vecteur coût $(46, 25^*, 42)$ est dominée par celle ayant le vecteur coût $(14^*, 25, 31)$. De plus, parmi ces deux solutions optimales, celle qui est non dominée est également une solution optimale pour le premier objectif. Nous pouvons initialiser l'ensemble des solutions (temporairement) efficaces E^0, dans l'espace des objectifs par l'ensemble des solutions optimales non dominées. Ainsi, initiallement, le front Pareto optimal est :

$$Z(E^0) = \{(14^*, 25, 31); (27, 54, 15^*)\}.$$

Chaque tâche doit être affectée à un ouvrier, il en coûtera au moins 1 pour que la tâche 1 soit assurée, au moins 2, 4 et 4 (correspondants aux éléments soulignés de la matrice C_1) pour les tâches 2, 3 et 4 respectivement.

La somme des minimums des colonnes de la matrice C_1 constitue une borne inférieure du coût de toute affectation réalisable (dans l'exemple ci-dessus, cette somme est égale à 1+2+4+4=11).

De manière similaire, nous calculons les sommes des minimums des colonnes des matrices C_2 et C_3 (égaux à 9+7+5+3=24 et 3+1+2+4=10 respectivement). Alors le vecteur (11, 24, 10) dont la k-ème composante, $(k = 1..m)$ représente la somme des minimums des colonnes de la matrice C_k, est une borne inférieure du vecteur coût de toute affectation réalisable. Notons également, qu'on peut améliorer cette évaluation par les valeurs du point idéal (voir Lemme 2.2).

Séparons l'ensemble des 24 affectations possibles, en les distinguant selon l'ouvrier assigné à la tâche T_1 et calculons l'évaluation (le vecteur minorant des bornes inférieures des trois fonctions objectifs) des affectations dans chacun des quatre groupes ainsi constitués.

Par exemple, toute affectation pour laquelle l'ouvrier O_1 effectue la tâche T_1 aura un vecteur coût au moins égal à (**20**, 30, **25**). Les éléments en gras représentent des bornes inférieures atteintes par la fonction objectif correspondante (évaluation exacte, donnée en une solution réalisable).

La solution $s'^* = (1, 3, 2, 4)$, avec $Z(s'^*) = (20, 39, 25)$, du sous ensemble S' des affectations pour lesquelles l'ouvrier O_1 est affecté à la tâche T_1, possède une évaluation exacte pour le premier et le troisième objectif. En effet, cette solution correspond au cas où chaque ligne de C_1 (respectivement de C_3) contient un élément minimum de sa colonne (voir la représentation suivante des matrices C_1, C_2 et C_3).

$$\begin{bmatrix} 1 & 2 & 19 & 10 \\ 18 & 19 & \underline{4} & 5 \\ 4 & \underline{11} & 17 & 12 \\ 2 & 14 & 19 & \underline{4} \end{bmatrix} \quad \begin{bmatrix} 15 & 7 & 6 & 18 \\ 15 & \underline{7} & 6 & 12 \\ 9 & 15 & 6 & 18 \\ 15 & 13 & \underline{5} & \underline{3} \end{bmatrix} \quad \begin{bmatrix} 18 & 17 & 13 & 9 \\ 11 & 17 & \underline{2} & 7 \\ 8 & \underline{1} & 11 & 13 \\ 3 & 7 & 15 & \underline{4} \end{bmatrix}$$

Les résultats pour les quatre groupes sont représentés par la figure 2.2.

L'arborescence sera explorée d'après l'ordre suivant lequel les nœuds sont crées. Et l'étiquette associée aux nœuds, dans le premier niveau de l'arborescence (figure 2.2) montre le numéro de l'ouvrier affecté à la tâche T_1.

Dans toutes nos figures, les notations suivantes sont utilisées :

- ($) : Solution efficace redondante.
- ($\pounds$) : Evaluation dominée par une solution temporairement efficace.
- (μ) : Evaluation exacte dominée par une solution temporairement efficace.
- (*) : Nœud stérilisé.

Aucune des affectations associées à la tâche T_1 dans la Figure 2.2, ne possèdent une évaluation dominée par une solution temporairement efficaces de $Z(E^0)$. Il s'ensuit que touts les nœuds du premier niveau (ouvriers affectés à la tâche T_1) seront séparés. Mainte-

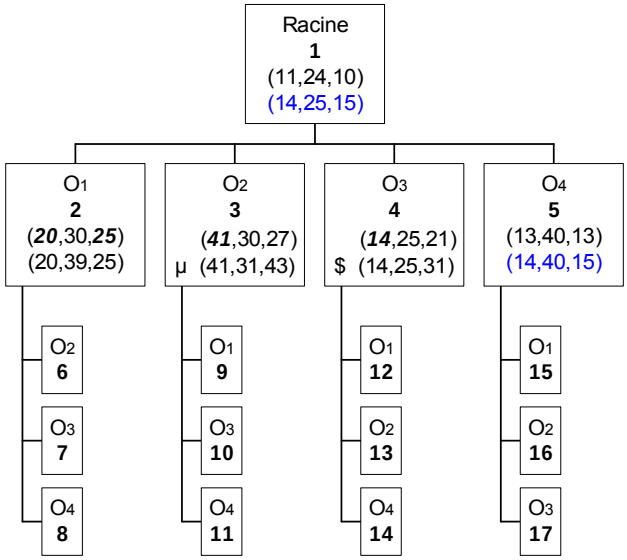


FIG. 2.2: Affectations associées à la tâche T_1

nant, si un nœud possède une évaluation exacte pour laquelle une affectation réalisable est trouvée, et si cette dernière n'est dominée par aucune des solutions appartenants au front Pareto optimal temporaire généré par l'algorithme, alors cette solution (son vecteur coût) est rajoutée à cet ensemble et inversement, toutes les solutions du front Pareto optimal temporaire, dominées par cette solution sont enlevées.

A la fin de la première étape, le front Pareto optimal temporaire $Z(E^1)$, correspondant aux affectations associées à la tâche T_1, est égal à $Z(E^0) \cup \{(20, 39, 25)\}$, i.e.

$$Z(E^1) = \{(14, 25, 31);\ (27, 54, 15) :\ (20, 39, 25)\}.$$

Dans l'étape 2, nous avons développé davantage l'ensemble des solutions (de la figure 2.3 à la figure 2.6).

En explorant tous les sommets induits par l'arborescence de la figure 2.2, on est assuré de parcourir toutes les solutions efficaces. Mais cette procédure est bien entendu grossièrement infructueuse puisque l'arborescence engendrée atteint très vite des proportions colossales la rendant inutilisable. Les tables table 2.2 à table 2.5 représentent l'énumération complète des affectations dans l'espace des objectifs.

Observons que dans la figure 2.3, lorsqu'on décide d'affecter l'ouvrier O_3 à la tâche T_2, l'évaluation est exacte pour deux objectifs sur les trois objectifs. On dispose ainsi d'une affectation réalisable dont la première et la troisième fonctions objectifs sont simultanément optimisées (dans le sous ensemble des affectations tel que l'ouvrier O_1 est affecté à

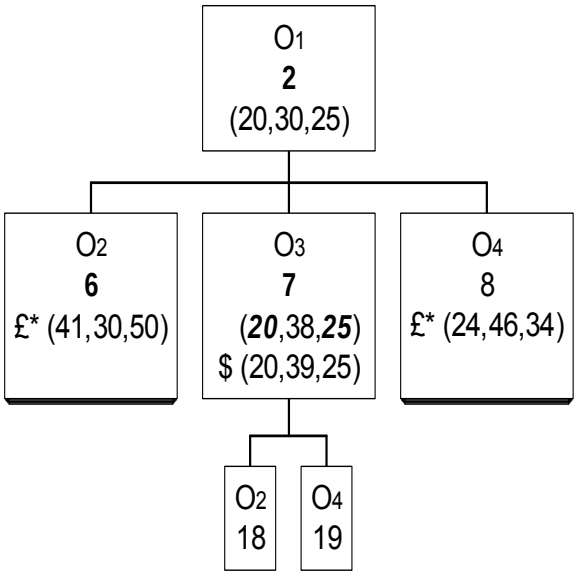


FIG. 2.3: Ouvrier O_1 affecté à la tâche T_1

N	Coûts des affectations
1	41 31 50
2	51 45 63
3	20 39 25
4	36 47 41
5	31 52 40
6	37 46 43

TAB. 2.2: Sous ensemble d'affectations pour lesquels l'ouvrier O_1 est affecté à la tâche T_1

N	Coûts des affectations
7	41 31 43
8	51 45 56
9	52 39 29
10	58 53 36
11	63 52 44
12	59 52 38

TAB. 2.3: Sous ensemble d'affectations pour lesquels l'ouvrier O_2 est affecté à la tâche T_1

N	Coûts des affectations
13	14 25 31
14	30 33 47
15	46 25 42
16	52 39 49
17	42 40 35
18	32 46 26

TAB. 2.4: Sous ensemble d’affectations pour lesquels l’ouvrier O_3 est affecté à la tâche T_1

N	Coûts des affectations
19	20 46 35
20	26 40 38
21	52 46 46
22	48 46 40
23	37 48 24
24	27 54 15

TAB. 2.5: Sous ensemble d’affectations pour lesquels l’ouvrier O_4 est affecté à la tâche T_1

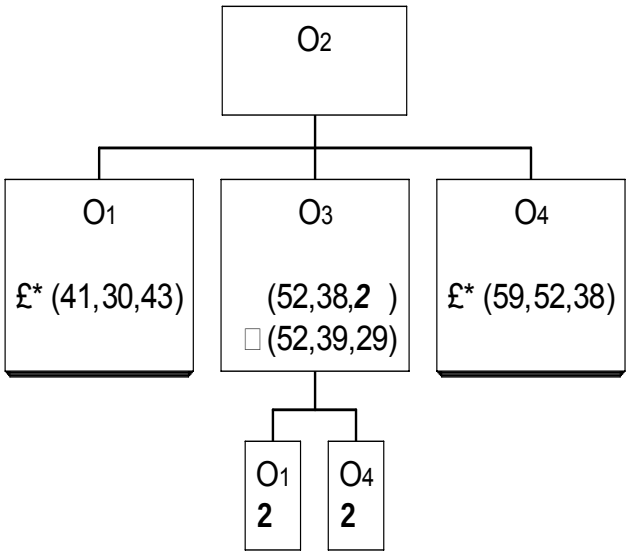


FIG. 2.4: Ouvrier O_2 affecté à la tâcheT_1

la tâche T_1 et l'ouvrier O_3 est affecté à la tâche T_2). De plus, pour les deux autres sous ensembles d'affectations (représentés par le premier et le troisième nœud de l'arborescence), on ne peut mieux faire, pour aucune des trois fonctions objectifs.

Ainsi, ces deux sous ensemble sont stérilisés.

A l'issue de la deuxième étape, le front Pareto optimal temporaire $Z(E^2)$, correspondant aux affectations associées à la tâche T_2, est égual à $Z(E^1)$ (i.e. $Z(E^2)$=$Z(E^1)$). Car toutes les affectations réalisables déterminées à cette étape sont ou bien dominées ou bien redondantes.

L'étape suivante correspond aux affectations associées à la tâche T_3.

A l'issue de la troisième et dernière étape, développée dans les figures figure 2.7 à la figure 2.10, nous obtenons :

$$Z(E^3) = Z(E^2) \cup \{(37, 48, 24)\}.$$

Ainsi, l'ensemble de toute les solutions efficaces de cette instance, dans l'espace des objectifs, construit par la méthode branch and bound proposée est :

$$Z(E) = \{(14, 25, 31);\ (27, 54, 15);\ (20, 39, 25);\ (37, 48, 24)\}.$$

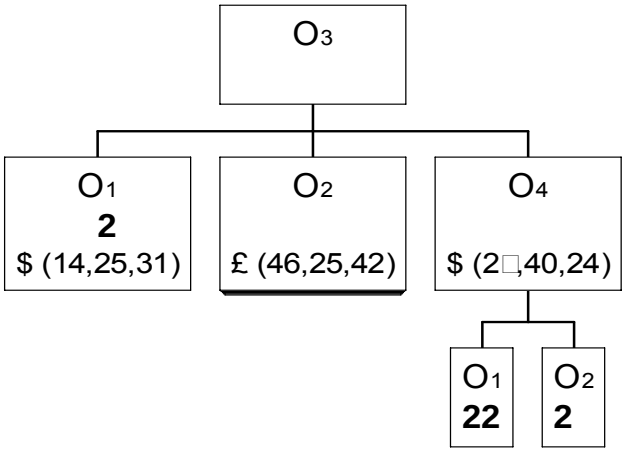


FIG. 2.5: Ouvrier O_3 affecté à la tâche T_1

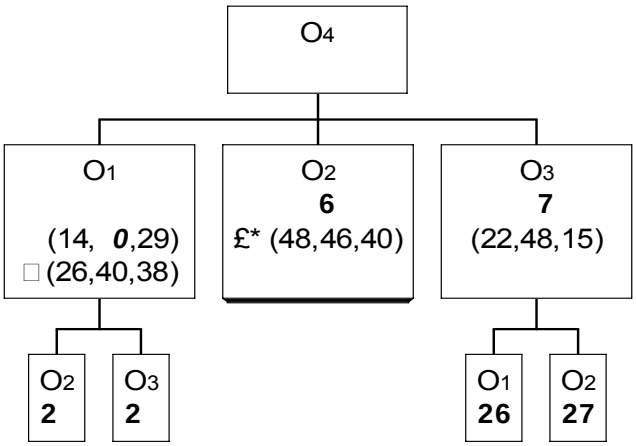


FIG. 2.6: Ouvrier O_4 affecté à la tâche T_1

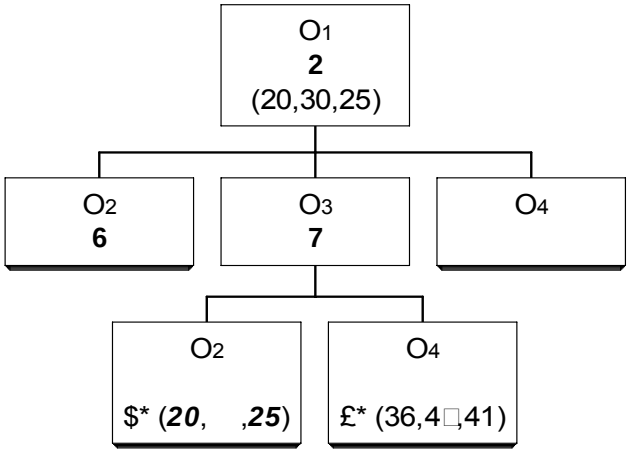


FIG. 2.7: Continuation de 2.3

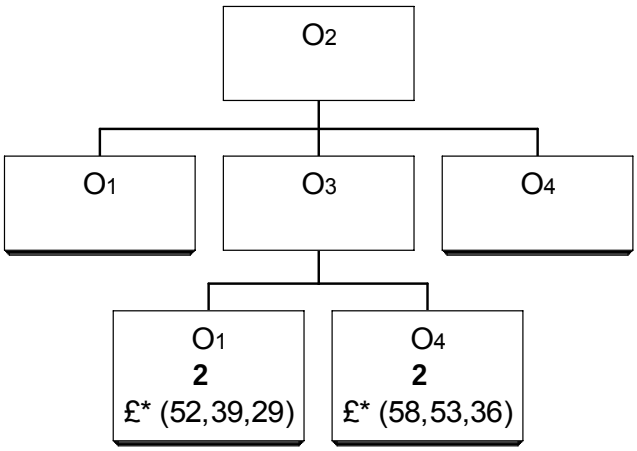


FIG. 2.8: Continuation de 2.4

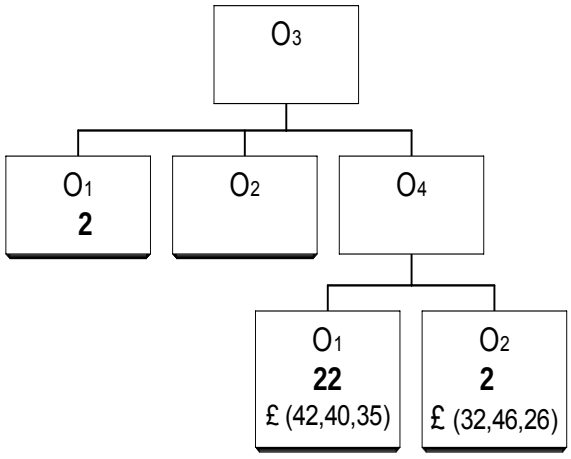


FIG. 2.9: Continuation de 2.5

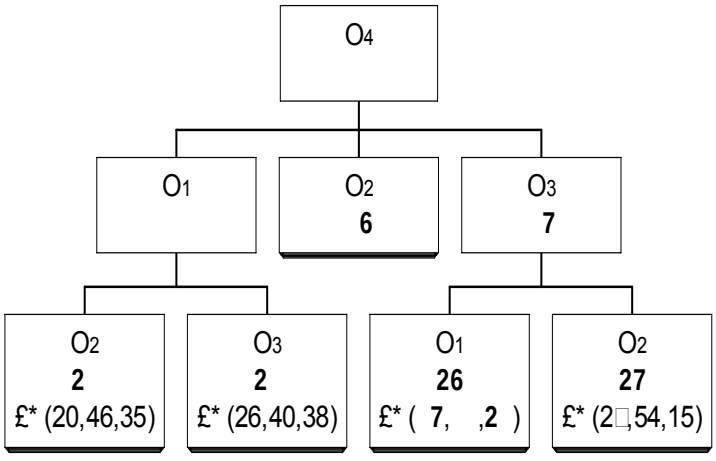


FIG. 2.10: Continuation de 2.6

2.6 Résultats expérimentaux

L'algorithme branch and bound présenté dans la section 2 est appliqué à l'espace complet des solutions pour déterminer l'ensemble E de toutes les affectations efficaces. Cet algorithme a été testé sur différentes tailles et nombre d'objectifs pour des problèmes affectation avec des matrices de coûts aléatoirement générées dans l'intervalle $[0, 20]$. Les résultats obtenus pour les instances générées sont montrés dans la table 2.6.

$m \times n$	% de nœuds visités	% de nœuds dominés	solutions efficaces	Cput time ($10^{-2}s$)
2×3	37.5	57.1	2	0
2×4	53.8	58.8	4	0
2×5	45.1	59.6	8	0
2×6	19.3	68.8	7	0
2×7	5.7	71.2	14	0
3×3	56.3	66.7	5	0
3×4	43.1	60.7	5	0
3×5	13.8	71.1	6	0
3×6	47.6	63.7	34	0
4×3	62.5	60.0	3	0
4×4	72.3	51.1	17	0
4×5	56.4	55.9	32	0
4×6	65.4	59.5	59	11

TAB. 2.6: Résultats expérimentaux sur quelques instances aléatoirement générées du PAMO

Contrairement à la décomposition systématique (énumération complète) du problème d'affectation en $n!$ affectations possibles, l'algorithme branch and bound proposé permet :

- d'éviter l'examen de certaines parties : en effet seulement 44.5% (en moyenne) de nœuds de l'arborescence ont été visités ;
- d'éliminer des branches entières : 61.9% (en moyenne) parmi les nœud visités ont été stérilisés.

2.7 Conclusion

Contrairement à la décomposition systématique du problème d'affectation en $n!$ affectations possibles, l'algorithme branch and bound proposé permet d'éviter l'examen de certaines parties, d'éliminer des branches entières et de déterminer l'ensemble E de toutes les affectations efficaces (supportées et non supportées). L'intérêt de notre méthode est qu'elle peut résoudre des problèmes d'affectations multi-objectifs avec trois objectifs et plus, contrairement à toutes les méthodes existantes qui sont limitées à la résolutions des problèmes d'affectation bi-objectif. De plus la méthode branch and bound proposée, conserve le contexte naturel multi-objectif du problème sans changer sa structure initiale.

En pratique, les méthodes branch and bound ne sont envisageables que pour des problèmes de petites tailles. Pour des nombreux problèmes, le nombre de solutions efficaces croît très vite avec la taille de l'instance à traiter (explosion combinatoire), ce qui engendre des temps de calcul rapidement prohibitifs (dépassant par exemple l'espérance de vie du système solaire). On se tourne alors vers des approches non exactes qui en des temps raisonnables, permettent de déterminer une “bonne” approximation de l'ensemble de solutions efficaces. Les métaheuristiques telles que le recuit simulé, la recherche tabou, les algorithmes génétiques, etc... fournissent des alternatives adéquates lorsque les méthodes exactes ne sont plus praticables. L'application de la métaheuristique du recuit simulé au problème d'affectation multi-objectif de grande taille fera l'objet de notre prochain chapitre.

Chapitre 3

Recuit simulé multi-objectif et relation de dominance

3.1 Introduction

Dans ce chapitre, nous nous intéressons à la métaheuristique du recuit simulé permettant d'approximer la frontière Pareto. Deux nouvelles approches [3] pour l'adaptation de la méthode du recuit simulé, au contexte multi-objectif, sont proposées. Ces approches seront décrites avec précision dans le cas du problème d'affectation multi-objectif et seront comparées au recuit simulé multi-objectif d'Ulungu [88]. Leurs performances seront illustrées sur des problèmes test standards [84].

3.2 Méthode du recuit simulé

La méthode du recuit simulé fait partie d'un groupe de méthodes dénommées *métaheuristiques*, comprenant notamment la méthode de recherche tabou, les algorithmes évolutionnaires, les algorithmes de colonies de fourmis, les réseaux de neurones, les essaims particulaires, ... apparues à partir des années 1980, avec une ambition commune : résoudre "au mieux" les problèmes d'optimisation difficiles (NP-complets). Ces méthodes sont inspirées par des analogies avec la physique (recuit simulé, diffusion simulée...), avec la biologie (algorithmes évolutionnaires, recherche tabou...) ou avec l'éthologie (colonie de fourmis, essaims particulaires,...) ; elles sont, au moins pour partie, stochastiques et partagent aussi les mêmes inconvénients relatifs aux difficultés de réglage des paramètres de la méthode et le temps de calcul élevé. De plus, ces méthodes ont marquées une réconciliation des deux domaines discret et continu : puisque celles-ci s'appliquent à toutes

sortes de problèmes discrets et peuvent s'adapter aussi aux problèmes continus. Elles sont "heuristiques" au sens où elles ne garantissent pas d'atteindre une solution optimale mais ont pour ambition de fournir une "bonne" solution approchée en un temps de calcul "raisonnable" ; l'ajout du préfixe "méta" signifie leur flexibilité d'application et d'adaptation à une large classe de problèmes. Le recuit simulé a été introduit pour la première fois par Kirkpatrik, Gelatt et Vecchi en 1983 [48] des spécialistes de physique statistique. Ils s'intéressaient aux configurations de basse énergie de matériaux magnétiques désordonnés, regroupés sous le terme de *verres de spin*. Des travaux semblables ont été développés indépendamment par Cerny en 1985. Et depuis son apparition, il a prouvé son efficacité dans divers domaines des sciences et de l'ingénieur tels que : la conception des circuits électroniques, le traitement des images, la collecte des ordures ménagères, l'organisation du réseau informatique, du loto, ...etc. Les méthodes du recuit simulé ont l'avantage d'être souples et rapidement implémentables lorsqu'on veut résoudre des problèmes d'optimisation combinatoire de grande taille.

3.2.1 Principe de la méthode

Le recuit simulé, utilisée dans la résolution des problèmes d'optimisation, s'inspire d'un phénomène naturel rencontré en métallurgie : un métal est porté à sa température de fusion (qui correspond à une énergie élevée) puis, il est refroidi le plus lentement possible, en marquant des paliers de température de durée suffisante ; si la descente en température est trop rapide, il apparaît des défauts qui peuvent être éliminés par réchauffement local. Cette stratégie de baisse contrôlée de température conduit à un état solide stable, correspondant à une configuration d'énergie minimum. La technique opposée est celle de la trempe, qui consiste à abaisser très rapidement la température du métal en fusion et on obtient ainsi un état métastable qui correspond à un minimum local de l'énergie. En optimisation, il suffit d'introduire un paramètre de contrôle qui joue le rôle de la température et de remplacer l'énergie du système par la fonction objectif. Le recuit simulé est une méthode probabiliste issue de l'algorithme de Métropolis [56] qui permet de générer aléatoirement une suite de configurations microscopiques d'atomes en interaction. Cette suite, obéit à la loi de *Bolzmann* sur l'équilibre d'un système thermodynamique de la physique statistique à une température donnée. En effet, d'après *Bolzmann* lorsque l'équilibre thermodynamique est atteint à une température donnée T, la probabilité, pour un système physique, de posséder une énergie donnée E, est proportionnel au facteur de *Bolzmann* : $exp(-\frac{E}{T})$. Ce qui signifie que les configurations d'énergies inférieures sont

favorisées. D'autre part, pour éviter d'être piégé en des optimums locaux, on accepte "parfois" (règle de Métropolis) des transitions vers des configurations qui détériorent l'énergie contrairement au principe de l'amélioration itérative (figure 3.1).

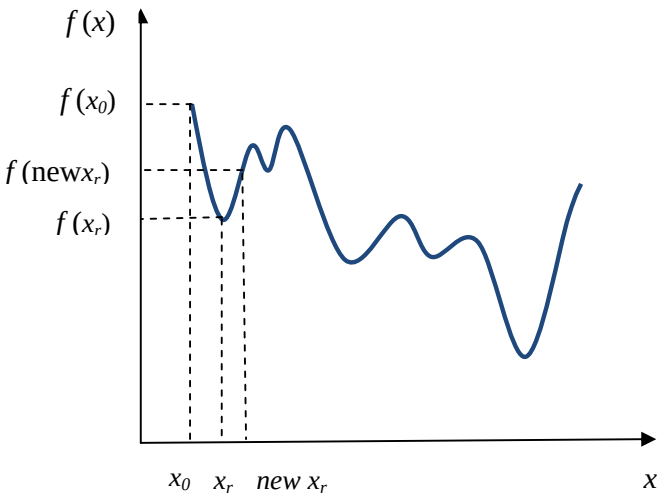


FIG. 3.1: Allure de la fonction objectif d'un problème d'optimisation difficile

3.2.2 Algorithme de la méthode

Etant donné un problème d'optimisation combinatoire mono-objectif du type :

$$(POC)\begin{cases} min\ f(x) \\ x \in S \end{cases}$$

L'exploration de l'ensemble des solutions réalisables S, par l'algorithme du recuit simulé, est effectuée de manière stochastique ce qui permet de faire face à l'explosion combinatoire des possibilités. Cette exploration, correspond à une marche aléatoire suivant un processus Markovien (en ce sens que chaque solution ne dépend que de celle qui la précède immédiatement). La transition d'une solution courante x vers une autre solution y (à la température courante T) est gérée par la matrice des probabilités de transition :

$$P_T(x,y) = \begin{cases} 0, & \text{si } y \text{ n'appartient pas au } V(x) \text{ et } y \neq x\,; \\ \frac{1}{|V(x)|} & \text{si } y \neq x, y \in V(x) \text{ et } \Delta f \leq 0\,; \\ \frac{1}{|V(x)|}.\exp(-\frac{\Delta f}{T}) & \text{si } y \neq x, y \in V(x) \text{ et } \Delta f > 0\,; \\ 1 - \sum_{z \neq x} P_T(x,z) & \text{si } y = x\,; \end{cases}$$

En effet, une solution y est sélectionné aléatoirement dans le voisinage $V(x)$ de x. Chaque $y \in V(x)$ possède une probabilité $\frac{1}{|V(x)|}$ d'être choisi. $\Delta f = f(y) - f(x)$ est calculé et si $\Delta f \leq 0$, alors y est accepté (cas 2 dans la définition de P_T). Sinon, une deuxième décision (relative à l'acceptation ou le rejet de la solution y) indépendante du choix de y, parmi les voisins possibles de x, est prise avec une probabilité $\frac{1}{|V(x)|}.\exp(-\frac{\Delta f}{T})$ (cas 3 dans la définition de P_T). Dans le dernier cas, la probabilité de maintenir x est obtenue par complémentarité à 1.

Deux hypothèses simples assurent que la chaîne de Markov (à une température donnée T) tend vers un équilibre :

Hypothèse 1 : Symétrie de la structure de voisinage, c'est à dire si $y \in V(x)$ alors $x \in V(y)$.

Hypothèse 2 : Connexité de la structure de voisinage, i.e. $\forall x, y \in V(x)$, $\exists x_1, \ldots, x_i, \ldots, x_r$ une suite telle que $x_1 = x$, $x_r = y$ et $x_{i+1} \in V(x_i)$, $\forall i = 1, \ldots, r-1$. En d'autres termes, il est possible d'atteindre n'importe quelles solutions de n'importe quelle autre solution en un nombre fini de voisins.

Sous ces deux hypothèses, l'évolution du processus Markovien, gérée par les probabilités de transition P_T, pour chaque température T, lorsque le nombre de transitions est illimité (en pratique lorsque la chaîne est de longueur suffisante), tend vers la distribution stationnaire (distribution de Boltzmann) :

$$P_T(x) = \frac{exp(-\frac{f(x)}{T})}{\sum_{z \in S} \exp\left(-\frac{f(z)}{T}\right)} \tag{3.1}$$

L'examen du comportement de la probabilité P_T montre que :

- Quand T décroît ($T \neq 0$), l'exploration aléatoire imposée par la distribution de Boltzmann favorise les solutions "d'énergie inférieure" d'autant plus que la température est basse. En effet P_T se concentre de plus en plus sur les solutions "d'énergie basse" (valeur de f proche du minimum).
- A la limite 0 ($T = 0$), il n'y a pas de transition possible vers une solution "d'énergie plus grande" $\exp(-\frac{f(x)}{T}) = 0$, est par conséquent, P_0 est concentré exclusivement sur les minimums globaux.

L'algorithme de la méthode peut être décrit succinctement comme suit : Soit x_0 une solution initiale arbitrairement choisie dans S. Notons x_r la solution à l'étape r ($r = 1, \ldots$). On choisit aléatoirement une solution y, dans le voisinage $V(x_r) \subset S$ de x_r puis on pose :

$\Delta f = f(y) - f(x_r)$. L'exploration aléatoire de l'ensemble des solutions réalisable S est effectuée suivant la règle de Métropolis suivante :

- Si $\Delta f \leqslant 0$, alors y a pour effet de diminuer la valeur de la fonction objectif et par conséquent y devient la nouvelle solution courante x_{r+1}.
- Si $\Delta f > 0$, alors y provoque au contraire une augmentation de la valeur de la fonction objectif. Elle est acceptée tout de même, avec la probabilité $P_{T(r)} = exp(-\frac{\Delta f}{T(r)})$, où $T(r)$ est la température, une fonction décroissante de r.

En pratique, cette condition est réalisée de la manière suivante : on tire un nombre au hasard dans $]0, 1[$ et on le compare à $exp(-\frac{\Delta f}{T(r)})$. Si ce nombre aléatoire est inférieur à $exp(-\frac{\Delta f}{T(r)})$, x devient la nouvelle solution courante, sinon x_r demeure la solution courante. On diminue progressivement la température et on obtient alors des solutions dont la valeur de la fonction objectif est globalement décroissante. Le schéma de décroissance de $T(r)$ généralement adopté est le suivant : après chaque série de N itérations, on multiplie la température précédente par une constante réelle δ strictement inférieure à 1 :

$$T(r) = T_k = \delta^k \times T_0 \tag{3.2}$$

pour $kN \leq r < (k+1)N$ et k entier. Dans la pratique, la procédure est stoppée lorsque, pour une certaine valeur très basse de T, plus aucune détérioration de la solution n'est acceptée (en effet la procédure devient de plus en plus sélective à mesure que le temps s'écoule) et / ou lorsque au bout d'un certain nombre fixé d'itérations, la fonction $f(y)$ n'a pas été améliorée. La solution retenue est alors la dernière solution courante.

3.2.3 Implémentation de la méthode

La mise en oeuvre de l'algorithme du recuit simulé ainsi que sa vitesse de convergence dépend essentiellement de la définition des éléments suivants :

- Structure de voisinage adaptée au problème.
- Température initiale T_0.
- Facteur de décroissance de la température δ.
- Nombre d'itérations par palier de température N.
- Critère d'arrêt.

L'équilibre recherché parmi ces paramètres est le résultat d'une phase expérimentale. Néanmoins, des suggestions pratiques et simples ont été proposées dans la littérature [73], pour fixer à priori ces paramètres de contrôle.

Température initiale T_0 : on peut la calculer au préalable suivant :

- A partir d'une solution initiale, faire 100 perturbations au hasard, évaluer la moyenne $\overline{\Delta E}$ des variations ΔE correspondantes.
- Choisir un taux initial d'acceptation τ_0 des "perturbations dégradantes", selon la qualité supposée de la solution initiale. Exemple

  qualité "médiocre" : $\tau_0 = 50\%$ (démarrage à haute température).

  qualité "bonne" : $\tau_0 = 20\%$ (démarrage à basse température).
- Déduire T_0 de la relation $\tau_0 = \exp(-\frac{\overline{\Delta E}}{T_0})$

- *Changement de palier de température* : peut s'opérer dés que l'une des deux conditions suivantes est satisfaite au cours du palier de température :
  - $12.n$ perturbations acceptées.
  - $100.n$ perturbations tentées.

  où n désigne la taille du problème.
- *Décroissance de la température* : peut être effectuée selon la loi géométrique : $T_{k+1} = 0.9 \times T_k$.
- *Critère d'arrêt* : peut être opéré après 3 paliers de température successifs sans aucune acceptation.

3.3 Recuit simulé multi-objectif d'Ulungu

Notons $\widehat{E}$: la liste de toutes les solutions générées durant la procédure du recuit simulé. Cette liste est actualisée au fur et à mesure de l'évolution de la méthode, en tenant compte des nouvelles solutions générées. Soit x_0 une solution initiale arbitrairement choisie dans S. Notons x_r la solution à l'étape r et soit y une solution générée aléatoirement dans le voisinage $V(x_r)$ de x_r. Du fait de la présence de plusieurs objectifs, la comparaison des solutions x_r et y est effectuée sur la base des écarts : $\Delta f^k = f^k(y) - f^k(x_r)$ pour $k = 1..m$

- Si $\forall k \in \{1, \ldots, m\}$ on a $\Delta f^k \leqslant 0$, alors y domine x_r. Par conséquent :
  - y devient la nouvelle solution courante x_{r+1}.
  - $\widehat{E}$ est mise à jour par comparaison par paire, de y avec toutes les solutions de $\widehat{E}$, seules les solutions non dominées par rapport aux solutions précédemment générées par la procédure sont retenues.
- Si $\exists k, k' \in \{1, \ldots, m\}$ tels que $\Delta f^k < 0$ et $\Delta f^{k'} > 0$: alors aucune des deux solutions y et x_r ne domine l'autre. Par conséquent :
  - y est choisie comme nouvelle solution courante x_{r+1}.

- $\widehat{E}$ est mise à jour par comparaison par paire avec y.

x_r aurait pu être choisie également comme nouvelle solution courante, puisque aucune des solutions y et x_r ne s'impose à l'autre. Néanmoins, il semble préférable de choisir systématiquement y comme nouvelle solution courante afin d'investiguer un autre voisinage et déterminer plus de solutions.

– Si $\forall k \in \{1, \ldots, m\}$ on a $\Delta f^k \geq 0$, alors y est dominée par x_r. En appliquant l'algorithme de Métropolis, la solution y peut néanmoins être sélectionnée avec une certaine probabilité $p \in]0, 1[$ choisie aléatoirement et comparée à une autre probabilité dépendant des écarts Δf^k. Le calcul de cette probabilité est effectué sur la base de la fonction d'agrégation U définie dans l'expression (1.11) (Chapitre 1). Posons :

$$\Delta U = \sum_{k=1}^{m} \lambda^k \Delta f^k; \quad m \geq 2. \tag{3.3}$$

- Si $p \leq \exp\left(-\frac{\Delta U}{T(r)}\right)$, alors y est néanmoins la nouvelle solution courante x_{r+1}.
- Si $p > \exp\left(-\frac{\Delta U}{T(r)}\right)$, alors x_r est conservée comme solution courante x_{r+1}.

L'algorithme suivant, décrit par Algorithme 2, permet de déterminer la liste $\widehat{E}(\lambda^j)$, des solutions potentiellement efficaces suivant la direction de recherche λ^j.

Remarques

- L'ensemble des itérations est généré comme dans un recuit simulé classique, à l'aide des paramètres T_0, δ, N_{STOP}, T_{STOP}.
- Le cardinal de la liste $\widehat{E}(\lambda^j)$ s'accroît au fur et à mesure de la progression de la procédure.
- Plusieurs autres fonctions d'agrégation (fonctions scalarisantes) peuvent être utilisées (Chapitre 1). Bien que le choix de l'un ou l'autre type de fonction scalarisante peut influencer le déroulement de la procédure, néanmoins son effet est limité par le caractère aléatoire de la méthode.

L'utilisation d'une fonction d'agrégation au sein du mécanisme du recuit simulé, va inévitablement privilégier la recherche suivant une certaine direction définie à priori. Pour palier à cet inconvénient, Ulungu a envisagé la diversification de la recherche aléatoire dans d'autres directions, en examinant plusieurs jeux de poids diversifiés. Notons :

– λ^j $(j = 1..J)$, les différentes directions diversifiées de recherche.
– $\widehat{E}(\lambda^j)$, la liste des solutions potentiellement efficaces obtenues suivant la direction de recherche λ^j.

Cependent, de nombreuses solutions de $\widehat{E}(\lambda^l)$ peuvent être dominées par des solutions engendrées à l'aide d'une autre direction λ^q $(q = 1..J,\ l = 1..J\ ;\ q \neq l)$ et figurant dans

Algorithme 2 Algorithme MOSA d'Ulungu

Entrées: Une liste initiale des solutions potentiellement efficaces.

Sorties: Une approximation de l'ensemble de solutions efficaces suivant la direction de recherche λ^j.

1: $\widehat{E}(\lambda^j) := \{x_0\}$, ensemble initial de solutions potentiellement efficaces
2: Initialiser la procédure du recuit simulé.
3: **répéter**
4: **répéter**
5: Générer aléatoirement une nouvelle solution $new\_x \in V(x)$
6: Calculer $\Delta Z^k = Z^k(new\_x) - Z^k(x), k = 1 \dots m$
7: **si** $(\forall k = 1 \dots m,\ \Delta Z^k \leq 0)$ ou $(\exists k, k'$ tels que $\Delta Z^k > 0$ et $\Delta Z^{k'} < 0)$ **alors**
8: $x := new\_x$
9: mise à jour de $\widehat{E}(\lambda^j)$
10: **sinon**
11: Générer aléatoirement une probabilité p
12: Calculer $\Delta U_j = U(new\_x, \lambda^j) - U(x, \lambda^j)$
13: **si** $p \leq \exp\left(-\frac{\Delta U_j}{T}\right)$ **alors**
14: $x := new\_x$
15: **finsi**
16: **finsi**
17: **jusqu'à** $nbr\_iter = N_{STOP}$
18: $T := \delta \times T$
19: **jusqu'à** $T \leq T_{STOP}$
20: $\widehat{E}(\lambda^j)$ est une approximation de l'ensemble de solutions efficaces suivant la direction λ^j.

$\widehat{E}(\lambda^q)$ et vice versa. C'est pourquoi, il serait nécessaire d'analyser l'ensemble

$$\bigcup_{j=1}^{J} \widehat{E}(\lambda^j)$$

à l'aide de comparaison par paires et seules les solutions non dominées seront en fin compte retenues.

3.3.1 Application de MOSA d'Ulungu au problème d'affectation bi-objectif

Dans cette section, nous appliquerons le recuit simulé multi-objectif d'Ulungu au problème d'affectation bi-objectif. Une phase initiale importante comporte :

- La définition de la structure de voisinage, exploitant la structure topologique spécifique du problème d'affectation.
- L'initialisation de l'algorithme de la méthode .

Problème d'affectation :

Rappel. Le problème d'affection bi-objectif revient à résoudre le programme linéaire :

$$(biAP)\begin{cases} \text{“ min ”} Z^k(x) = \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}^k x_{ij} & k = 1..2 \\ \displaystyle\sum_{j=1}^{n} x_{ij} = 1 & i = 1..n \\ \displaystyle\sum_{i=1}^{n} x_{ij} = 1 & j = 1..n \\ x_{ij} \in \{0,1\} & i = 1..n,\ j = 1..n \end{cases}$$

De façon équivalente le problème revient à trouver une permutation de n objets, de coût minimal. Si P représente l'ensemble des permutations de n objets, dans ce cas le problème peut se formuler :

$$\min_{\rho \in P} \sum_{i=1}^{n} c_{i\rho(i)} \tag{3.4}$$

Structure de voisinage

Soit $Affect$ l'affectation courante. Notons new_Affect la nouvelle affectation générée dans le voisinage de $Affect$ ($V(Affect)$) dont la structure est construite d'après le schéma suivant :

$$new\_Affect(i) = \begin{cases} Affect(i), & \text{si } i \neq i_1, i_2\,; \\ Affect(i_2) & \text{si } i = i_1\,; \\ Affect(i_1) & \text{si } i = i_2\,; \end{cases}$$

Le voisinage $V(Affect)$ est formé de toutes les affectations obtenues en permutant deux composantes de l'affectation courante et en maintenant les $(n-2)$ autres composantes inchangées.

Initialisation de algorithme

Une manière simple de construire une solution initiale, $Affect_0$, consiste soit à générer aléatoirement une permutation de n objets soit à utiliser l'algorithme glouton [84] selon une direction de recherche λ^j pour $j \in \{1, \ldots, J\}$, choisie aléatoirement parmi les J directions de recherche. Une solution initiale peut être aussi calculée, en attribuant aux objectifs des poids, puis en appliquant au problème d'affectation mono-objectif ainsi obtenu la méthode hongroise.

La procédure d'Ulungu, met l'accent sur la nécessité :

- d'utiliser des directions de recherche diversifiées λ^j, $j = 1, \ldots, J$ au moyen d'un algorithme récursif de type "bisection" ([85]) qui détermine un nombre minimale J de direction de recherche susceptibles de générer une très bonne approximation de la frontière Pareto.
- de trier les solutions des diverses listes $\widehat{E}(\lambda^j)$.

Pour illustrer cette procédure, considérons le problème d'affectation bi-objectif suivant :

Exemple 1 : $n = 7$

$$C_1 = \begin{bmatrix} 6 & 16 & 7 & 16 & 10 & 13 & 19 \\ 16 & 3 & 9 & 2 & 10 & 7 & 19 \\ 10 & 5 & 13 & 12 & 10 & 1 & 12 \\ 13 & 5 & 17 & 6 & 15 & 17 & 19 \\ 15 & 19 & 10 & 9 & 11 & 8 & 13 \\ 11 & 19 & 19 & 12 & 15 & 19 & 13 \\ 7 & 7 & 3 & 14 & 8 & 5 & 6 \end{bmatrix}$$

$$C_2 = \begin{bmatrix} 18 & 12 & 6 & 19 & 10 & 0 & 19 \\ 17 & 11 & 13 & 16 & 12 & 10 & 7 \\ 2 & 11 & 8 & 0 & 6 & 3 & 5 \\ 8 & 10 & 12 & 0 & 15 & 12 & 10 \\ 9 & 18 & 0 & 13 & 6 & 3 & 8 \\ 7 & 3 & 2 & 18 & 19 & 10 & 9 \\ 2 & 4 & 12 & 3 & 5 & 19 & 11 \end{bmatrix}$$

L'ensemble des solutions efficaces de l'exemple 1 est représenté par la figure 3.2 dans l'espace des objectifs. Il y a 14 solutions efficaces, 7 supportées er 7 non supportées.

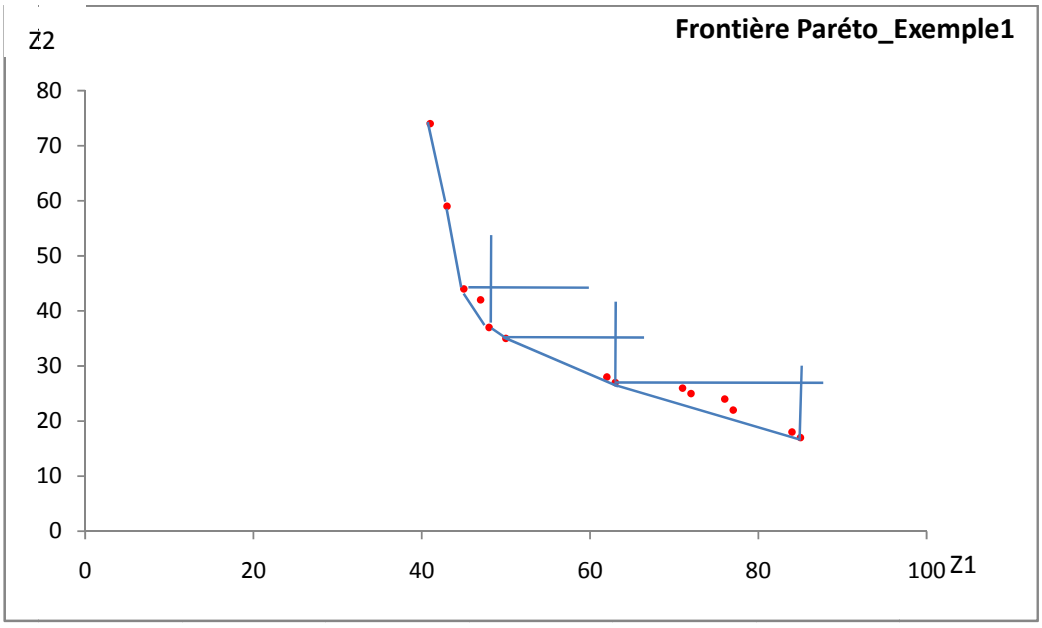


FIG. 3.2: Frontière_Paréto_2AP7

La figure 3.3 représente l'ensemble $\bigcup_{j=1}^{11} \widehat{E}(\lambda^j)$ des solutions obtenues avec 11 directions de recherche (Table 3.1).

	λ	$1-\lambda$
1	0	1
2	0.1	0.9
3	0.2	0.8
4	0.3	0.7
5	0.4	0.6
6	0.5	0.7
7	0.6	0.4
8	0.7	0.3
9	0.8	0.2
10	0.9	0.1
11	1	0

TAB. 3.1: Directions de recherche

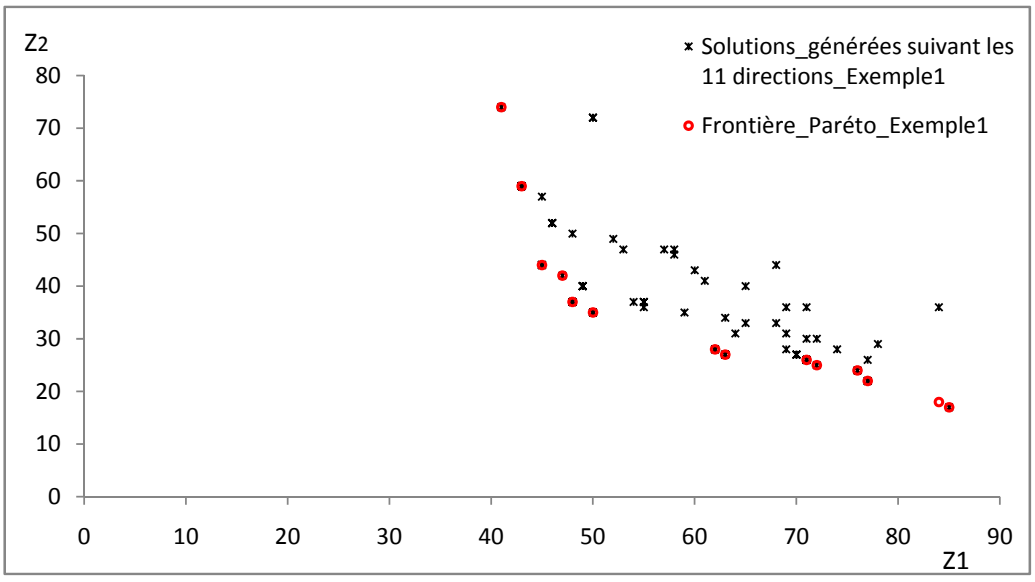


FIG. 3.3: Solutions_générées_Exemple1

Cet ensemble contient 88 solutions réparties de la manière suivante :

$$\begin{cases} |\widehat{E}(\lambda^1)| = 12 \\ |\widehat{E}(\lambda^2)| = 12 \\ |\widehat{E}(\lambda^3)| = 8 \\ |\widehat{E}(\lambda^4)| = 6 \\ |\widehat{E}(\lambda^5)| = 9 \\ |\widehat{E}(\lambda^6)| = 7 \\ |\widehat{E}(\lambda^7)| = 7 \\ |\widehat{E}(\lambda^8)| = 8 \\ |\widehat{E}(\lambda^9)| = 7 \\ |\widehat{E}(\lambda^{10})| = 6 \\ |\widehat{E}(\lambda^{11})| = 6 \end{cases}$$

La figure 3.4 contient les solutions de $\widehat{E}$ (obtenu après comparaison par paires des solutions de $\bigcup_{j=1}^{11} \widehat{E}(\lambda^j)$).

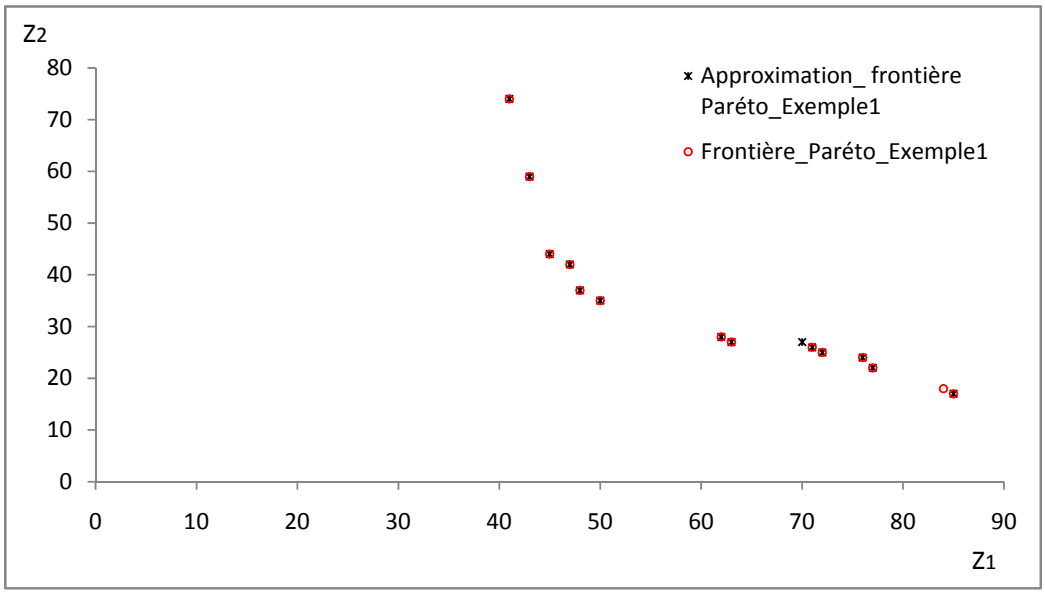


FIG. 3.4: Solutions_potentiellement_efficaces

Cet ensemble contient 15 solutions réparties de la manière suivante :

$$\begin{cases} |\widehat{E}(\lambda^1) \cap \widehat{E}| = 6 \\ |\widehat{E}(\lambda^2) \cap \widehat{E}| = 10 \\ |\widehat{E}(\lambda^3) \cap \widehat{E}| = 2 \\ |\widehat{E}(\lambda^4) \cap \widehat{E}| = 4 \\ |\widehat{E}(\lambda^5) \cap \widehat{E}| = 2 \\ |\widehat{E}(\lambda^6) \cap \widehat{E}| = 6 \\ |\widehat{E}(\lambda^7) \cap \widehat{E}| = 3 \\ |\widehat{E}(\lambda^8) \cap \widehat{E}| = 5 \\ |\widehat{E}(\lambda^9) \cap \widehat{E}| = 2 \\ |\widehat{E}(\lambda^{10}) \cap \widehat{E}| = 3 \\ |\widehat{E}(\lambda^{11}) \cap \widehat{E}| = 4 \end{cases}$$

3.4 Recuit simulé multi-objectif et relation de dominance

L'approche traditionnelle d'Ulungu, permet de générer une suite d'approximations des ensembles de solutions potentiellement efficaces suivant différentes directions de recherche (diversifiées), et de les fusionner, puis de débarrasser l'ensemble produit des solutions dominées. Dans cette section, nous proposons deux approches pour l'adaptation de la méthode du recuit simulé au contexte multi-objectif.

- Contrairement à l'approche traditionnelle d'Ulungu (approximation en aval), notre première approche consiste à agir en amont. La comparaison entre deux solutions voisines ainsi que la stratégie d'acceptation ou de rejet d'une solution de moindre qualité (dominée) sont définies à partir d'une relation de dominance utilisant toutes les directions de recherche en même temps.
- Dans la deuxième approche nous éliminons complètement la notion d'agrégation, et introduisons dans la règle de Métropolis une relation de dominance sur le vecteur des probabilités d'acceptation d'une solution dominée.

3.4.1 Dominance sur le vecteur utilité (DUSA (θ))

A chacune des solutions générées, au cours de la procédure du recuit, nous associons un nombre fini d'utilités. Considérons J directions de recherche (diversifiées) : $\lambda^j = (\lambda^{j_1}, \lambda^{j_2}, ..., \lambda^{j_m}), j = 1..J$. Nous obtenons le vecteur $(U_1(x), U_2(x), ..., U_J(x))$, repré-

sentant les utilités multiples de la solution x suivant les différentes directions de recherche $\lambda^1, \lambda^2, ..., \lambda^J$.

Considérons une solution courante x_r à l'étape r et soit $y \in V(x_r)$, une solution générée aléatoirement dans le voisinage $V(x_r)$ de x_r. Du fait que nous considérons simultanément les différentes directions de recherche, $\lambda^j = (\lambda^{j_1}, \lambda^{j_2}, \ldots, \lambda^{j_m})$, $j = 1..J$, la comparaison des solutions x_r et y est effectuée sur la base des écarts $(\Delta U_1, \Delta U_2, ..., \Delta U_J)$ des deux vecteurs d'utilité $U(y)$ et $U(x_r)$:

$$\Delta U_j = U(y, \lambda^j) - U(x_r, \lambda^j),\ j = 1..J. \tag{3.5}$$

Trois cas sont possibles :

Case 1 : $\Delta U_j \leq 0$ (avec au moins une inégalité stricte) alors y domine x_r suivant toutes les directions de recherche ;

Case 2 : $\Delta U_j \geq 0$ (avec au moins une inégalité stricte) alors y est dominée par x_r suivant toutes les directions de recherche ;

Case 3 : $\exists j, j' \in \{1, ..., J\}$ tels que $\Delta U_j < 0$ et $\Delta U_{j'} > 0$ alors aucune des deux solutions x_r et y ne domine l'autre (suivant certaines directions de recherche, y domine x_r et suivant d'autres directions de recherche, y est dominée par x_r).

La stratégie de sélection de la nouvelle solution courante consiste en les décisions suivantes :

- Si (cas 1 ou cas 3) alors $x_{r+1} := y$.
- Si cas 2 alors
  - $x_{r+1} := y$ avec une probabilité p
  - $x_{n+1} := x_r$ avec une probabilité $1 - p$

  où $p = \exp(-\frac{\theta(\Delta U)}{T})$ et θ une fonction réelle à plusieurs variables définie sur l'espace $\mathbb{R}^J$ des vecteurs ΔU.

Notons $\jmath = \{j \in \{1, ..., J\} \mid \Delta U_j > 0\}$

Nous définissons la fonction θ des quatre manières différentes suivantes :

- Première variante :

$$\theta_1(\Delta U) = \min_{j \in \jmath}\{\Delta U_j\}. \tag{3.6}$$

  La fonction θ_1 retient la plus petite dégradation parmi les dégradations obtenues suivant les différentes directions de recherche.

Algorithme 3 Approximation de E par la méthode (DUSA(θ))

Entrées: Une liste initiale des solutions potentiellement efficaces.

Sorties: Une approximation de l'ensemble de solutions efficaces.

$\widehat{E} := \{x_0\}$, ensemble initial de solutions potentiellement efficaces
Initialiser la procédure du recuit simulé.
répéter
  Générer uniformément J directions de recherche
  répéter
    Générer aléatoirement une nouvelle solution $new\_x \in V(x)$
    Calculer $\Delta U_j = U(new\_x, \lambda^j) - U(x, \lambda^j), j = 1..J$
    si $(\forall j = 1..J,\ \Delta U_j \leq 0)$ ou $(\exists j,\ j'$ tels que $\Delta U_j > 0$ et $\Delta U_{j'} < 0)$ **alors**
      $x := new\_x$
      mise à jour de $\widehat{E}$
    sinon
      Générer aléatoirement une probabilité p
      Calculer la fonction θ d'après les relations (3.6), (3.7), (3.8) ou (3.9)
      si $p \leq \exp(-\frac{\theta(\Delta U)}{T})$ **alors**
        $x := new\_x$
      finsi
    finsi
  jusqu'à N_{STOP}
  $T := \delta \times T$
jusqu'à $T \leq T_{STOP}$
$\widehat{E}$ est une approximation de l'ensemble de solutions efficaces.

– Seconde variante :

$$\theta_2(\Delta U) = \frac{1}{|\jmath|} \sum_{j \in \jmath} \Delta U_j. \tag{3.7}$$

La fonction θ_2 retient la moyenne arithmétique de toutes les dégradations obtenues suivant les différentes directions de recherche.

– Troisième variante :

$$\theta_3(\Delta U) = \max_{j \in \jmath} \{\Delta U_j\}. \tag{3.8}$$

La fonction θ_3 retient la plus mauvaise dégradation parmi les dégradations obtenues suivant les différentes directions de recherche.

– quatrième variante :

$$\theta_4(\Delta U) = \sum_{j \in \jmath} \Delta U_j. \tag{3.9}$$

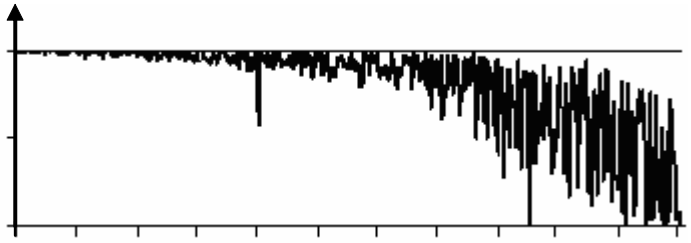
La fonction θ_4 retient la somme de toutes les dégradations obtenues suivant les différentes directions de recherche.

Nous étudions, dans un premier temps, l'impact du choix de la fonction θ, sur la probabilité d'acceptation d'une solution dominée. Pour cela, nous allons examiner l'évolution de cette probabilité au cours de la progression de la procédure du recuit simulé utilisant la relation de dominance sur la fonction utilité (Algorithme 3) pour chacune des fonctions $\theta_i, i = 1..4$ proposées. Posons :

– $e_{min} = \exp(-\frac{\theta_1(\Delta U)}{T})$,
– $e_{moy} = \exp(-\frac{\theta_2(\Delta U)}{T})$,
– $e_{max} = \exp(-\frac{\theta_3(\Delta U)}{T})$,
– $e_{som} = \exp(-\frac{\theta_4(\Delta U)}{T})$.

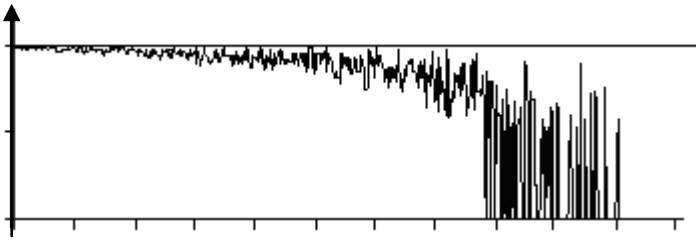
Nous avons appliqué Algorithme 3 à une instance du problème d'affectation bi-objectif de taille 40 et dont les coefficients des matrices coût sont uniformément distribués dans l'intervalle $[0, 20]$. Les résultats de l'évolution de la probabilité d'acceptation d'une solution dominée en fonction de $\theta_i, i = 1..4$, sont représentés dans les figures suivantes :

– Les différentes figures (figure 3.5, figure 3.6, figure 3.7 et figure 3.8) montrent que la probabilité d'acceptation converge vers zéro lorsque la température tend vers zéro .
– Maintenant, pour une température, même très basse, la probabilité d'acceptation, après avoir diminué de manière très sensible, peut augmenter de façon considérable si la dégradation de la solution générée par rapport à la solution courante est peut significative (qualité très proche de celle de la solution courante).



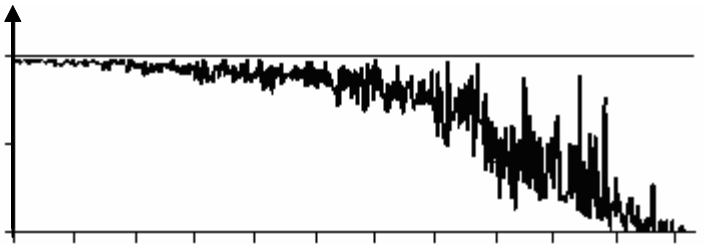
e e e

FIG. 3.5: Probabilité d'acceptation (2AP40)e_{min}



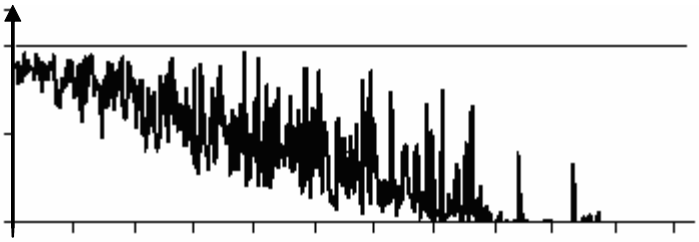
e e e

FIG. 3.6: Probabilité d'acceptation (2AP40)e_{moy}



e e e

FIG. 3.7: Probabilité d'acceptation (2AP40)e_{max}



e e e

FIG. 3.8: Probabilité d'acceptation (2AP40)e_{som}

– Le niveau de la probabilité d'acceptation diminue lorsque l'ordre de la dégradation augmente (min, moy, max, et som). Par exemple, dans le cas de la somme (θ_4), le niveau de la probabilité d'acceptation est le plus bas et par conséquent la procédure devient beaucoup plus sélective que dans les autres cas. .

3.4.2 Dominance sur le vecteur coût (DCSA)

La comparaison entre les solutions x_r et $y \in V(x_r)$ est effectuée à partir de la relation de dominance sur leur vecteurs coût associés : $(Z^1(x_r), \ldots, Z^m(x_r))$ et $(Z^1(y), \ldots, Z^m(y))$. De manière similaire à la relation de dominance sur le vecteur utilité, trois situations sont possibles :

Cas 1 : $\Delta Z^k \leq 0$ (avec au moins une inégalité stricte) alors y domine x_r suivant tous les objectifs ;

Cas 2 : $\Delta Z^k \geq 0$ (avec au moins une inégalité stricte) alors y est dominée par x_r suivant tous les objectifs ;

Cas 3 : $\exists k, k' \in \{1, ..., m\}$ tels que $\Delta Z^k < 0$ et $\Delta Z^{k'} > 0$ alors aucune des deux solutions x_r et y ne domine l'autre. Pour certains objectifs, y est préférée à x_r et pour d'autres objectifs c'est plutôt x_r qui est préférée à y.

Au lieu de faire une agrégation des écarts $\Delta Z^k, k = 1..m$, suivant une direction de recherche générée aléatoirement (Sérafini [71]) ou prédéfinie (Ulungu [88]), nous proposons d'éliminer complètement la notion d'agrégation. La sélection d'une solution dominée, dans la règle de Métropolis, est effectuée dans ce cas, à partir de la relation de dominance sur le vecteur des probabilités de la manière suivante :

Considérons le vecteur $e = (e_1, \ldots, e_m)$ des probabilités $e_k = \exp(-\frac{\Delta Z^k}{T})$ (d'acceptation ou de rejet d'une solution de moindre qualité) associées aux différents objectifs $Z^k, k \in \{1, ..., m\}$.
Et générons aléatoirement, suivant la loi uniforme, un vecteur $p = (p_1, \ldots, p_m)$ ($p_k \sim U(0,1), k = 1, \ldots, m$)).

De façon similaire, nous définissons une relation de dominance sur les vecteurs des probabilités de la manière suivante :

Cas 1 : $p_k \leq e_k$ (avec au moins une inégalité stricte) alors e domine p ;

Cas 2 : $p_k \geq e_k$ (avec au moins une inégalité stricte) alors e est dominée par p ;

Cas 3 : $\exists k, k' \in \{1, ..., m\}$ tels que $p_k < e_k$ et $p_k > e_k$ alors aucun des deux vecteurs p et e ne domine l'autre.

La règle de sélection d'une solution dominée, pour la diversification de la recherche dans d'autres voisinage, est alors la suivante :

- si (cas 1 ou cas 3) alors $x_{r+1} := y$.
- sinon $x_{r+1} := x_r$.

3.5 Résultats expérimentaux

Les performances des différentes variantes sont évaluées suivant deux critères :

- La qualité des résultats obtenus en calculant leur distance moyenne par rapport au point idéal : $d(\widehat{E}, x^*) = \min_{\{x \in \widehat{E}\}} d(x, x^*)$ où : $d(x, x^*) = \frac{1}{2} \sum_{k=1}^{2} |Z^k(x) - Z^k(x^*)|$ (voir Czyzak et Jaszkiweicz 1998 et Ulungu et al., 1998).
- Le temps d'exécution représenté par CPU (t) et mesuré en centième de seconde. (cput = 360000.h + 6000. mn +100. se + ce)

Tous les tests ont été réalisés sur un Pentium 4, 1.7 GHz, 256 MB, 20 GB. Les performances de nos approches sont illustrées sur les instances 2AP5, 2AP10, ..., 2AP40, du problème d'affectation bi-objectif générées par Gandibleux (www.terry.uga.edu mcdm) :

- Les coefficients des matrices "coût" sont uniformément générés dans l'intervalle [0,20].
- Taille des instances considérées varie entre 5×5 à 40×40.

Pour chaque instance, et pour chaque variante, dix tests seront réalisés et seules les valeurs minimales, moyennes et maximales (aussi bien pour la distance que pour le temps d'exécution) seront rassemblées sur un même tableau.

Algorithme 4 Approximation de E par la méthode DCSA

Entrées: Une liste initiale des solutions potentiellement efficaces.

Sorties: Une approximation de l'ensemble de solutions efficaces.

$\widehat{E} := \{x_0\}$, ensemble initial de solutions potentiellement efficaces
Initialiser la procédure du recuit simulé.
répéter
  répéter
    Générer aléatoirement une nouvelle solution $new\_x \in V(x)$
    Calculer $\Delta Z^k = Z^k(new\_xt) - Z^k(x), k = 1 \dots m$
    si $(\forall k = 1 \dots m,\ \Delta Z^k \leq 0)$ ou $(\exists k, k'$ tels que $\Delta Z^k > 0$ et $\Delta Z^{k'} < 0)$ **alors**
      $x := new\_x$
      mise à jour de $\widehat{E}$
    sinon
      Générer aléatoirement un vecteur probabilité $p = (p_1, p_2)$
      si $(\forall k = 1 \dots m,\ p_k \leq e_k)$ or $(\exists k, k'$ tels que $p_k < e_k$ et $p_k > e_k)$ **alors**
        $x := new\_xt$
      finsi
    finsi
  jusqu'à $nbr\_iter = N_{STOP}$
  $T := \delta \times T$
jusqu'à $T \leq T_{STOP}$
$\widehat{E}$ est une approximation de l'ensemble de solutions efficaces.

La fixation des paramètres du recuit simulé est le résultat d'une phase expérimentale. Plusieurs essais ont été réalisés sur les problèmes mono-objectifs relatifs aux différents objectifs, du problème multi-objectif considéré, dont les solutions optimales obtenues via la méthode hongroise, sont connues. Les paramètres du recuit simulé retenus sont :

$$N = 250,\ \alpha = 0.975, \quad T_0 = 1000, \quad T_{stop} = 0.1, \quad J = 11.$$

Les résultats numériques des différentes variantes du recuit simulé multi-objectif, sont reportés dans les tableaux suivant :

Instance	2AP5	2AP10	2AP15	2AP20	2AP25	2AP30	2AP40
Point idéal	(27,9)	(19,20)	(17,32)	(20,25)	(22,19)	(12,18)	(15,9)
Distance							
min	12	17	54.5	59.5	80	99.5	147
moy	12	19	60	63.55	93.2	113.9	150.41
max	12	22.5	66.5	75.5	100	123	151.5
CPUt (ce)							
min	21305	21335	22443	22547	25551	26188	27364
moy	21364.2	21479	23248	23535	25903	26316	27678
max	21514	21668	23991	25551	26337	26529	27957

TAB. 3.2: Résultats de la méthode MOSA d'Ulungu

Instance	2AP5	2AP10	2AP15	2AP20	2AP25	2AP30	2AP40
Point idéal	(27,9)	(19,20)	(17,32)	(20,25)	(22,19)	(12,18)	(15,9)
Distance							
min	12	17	59	53	81	74.5	125.5
moy	12	23.6	61.9	61.7	91.3	95.8	146.8
max	12	27	73.5	74.5	101	118	151.5
CPUt (ce)							
min	1829	2038	2428	2505	2598	2602	3026
moy	1856	2086	2483	2578	2723	2724	3109
max	1889	2137	2582	2686	2879	2883	3158

TAB. 3.3: Résultats de la méthode DUSA(min)

Instance	2AP5	2AP10	2AP15	2AP20	2AP25	2AP30	2AP40
Point idéal	(27,9)	(19,20)	(17,32)	(20,25)	(22,19)	(12,18)	(15,9)
Distance							
min	12	20	61	73	74.5	86.5	128
moy	12	25.2	70.6	79.6	93.2	102	148.4
max	12	27	73.5	83.5	101	121	151.5
CPUt (ce)							
min	1752	1856	2257	2351	2488	2543	2757
moy	1727.4	1834	2267.9	2484	2545	2634	2755
max	1912	1983	2637	2669	2702	2813	3207

TAB. 3.4: Résultats de la méthode DUSA(moy)

Instance	2AP5	2AP10	2AP15	2AP20	2AP25	2AP30	2AP40
Point idéal	(27,9)	(19,20)	(17,32)	(20,25)	(22,19)	(12,18)	(15,9)
Distance							
min	12	20	51.5	47	85.5	93	120
moy	12	23	67.9	62.4	93.8	98.9	144.1
max	12	25	73.5	77	101	110	151.5
CPUt (ce)							
min	1868	1895	2324	2378	2527	2598	3016
moy	1945	1981	2523	2554	2570	2630	2974
max	2033	2048	2648	2730	2806	2895	3729

TAB. 3.5: Résultats de la méthode DUSA(max)

Instance	2AP5	2AP10	2AP15	2AP20	2AP25	2AP30	2AP40
Point idéal	(27,9)	(19,20)	(17,32)	(20,25)	(22,19)	(12,18)	(15,9)
Distance							
min	12	19	61	63	76	83	93
moy	12	22	67.4	72.9	93.7	108.6	140
max	12	27	73.5	83.5	101	120.5	151.5
CPUt (ce)							
min	1307	1483	1988	2005	2054	2290	2532
moy	1472	1569	2219	2153.7	2340	2463	2667.7
max	1599	1675	2399	2427	2680	2718	2763

TAB. 3.6: Résultats de la méthode DCSA (dominance)

Les représentations graphiques des tendances moyennes des différents résultats sont données par les figures suivantes :

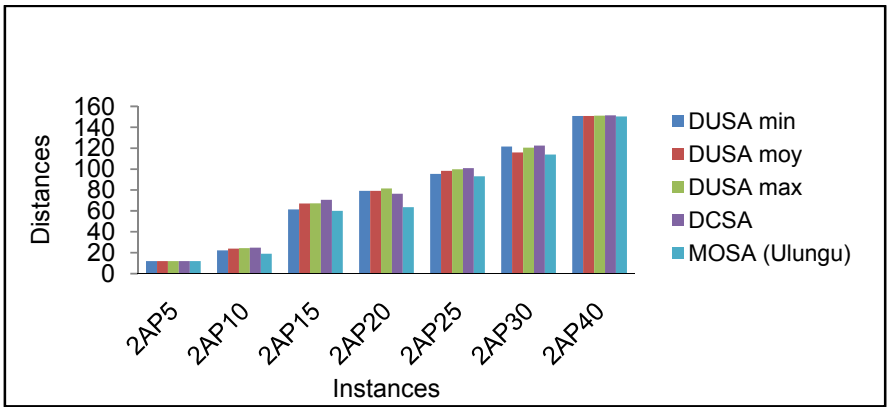


FIG. 3.9: Distance moyenne par rapport au point idéal

- Les résultats obtenus par la méthode MOSA d'Ulungu basée sur l'agrégation, sont légèrement meilleurs que ceux obtenus par nos différentes variantes basées sur la relation de dominance. Toute fois, les résultats dans les deux cas de figure (agrégation et dominance) restent de qualités comparables (figure 3.9).
- Par contre, les différentes approches Pareto proposées pour le recuit simulé sont en moyenne dix fois plus rapides que l'approche MOSA d'Ulungu (figure 3.10).
- Moins la procedure est sélective, mieux sont les résultats (voir la figure 3.9). En effet, les meilleurs simulations sont obtenues dans le cas où la probabilité d'acceptation

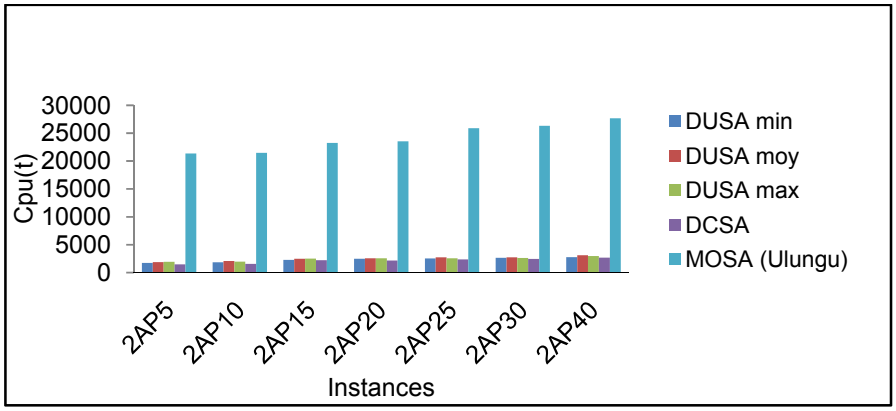


FIG. 3.10: Temps moyen d'exécution

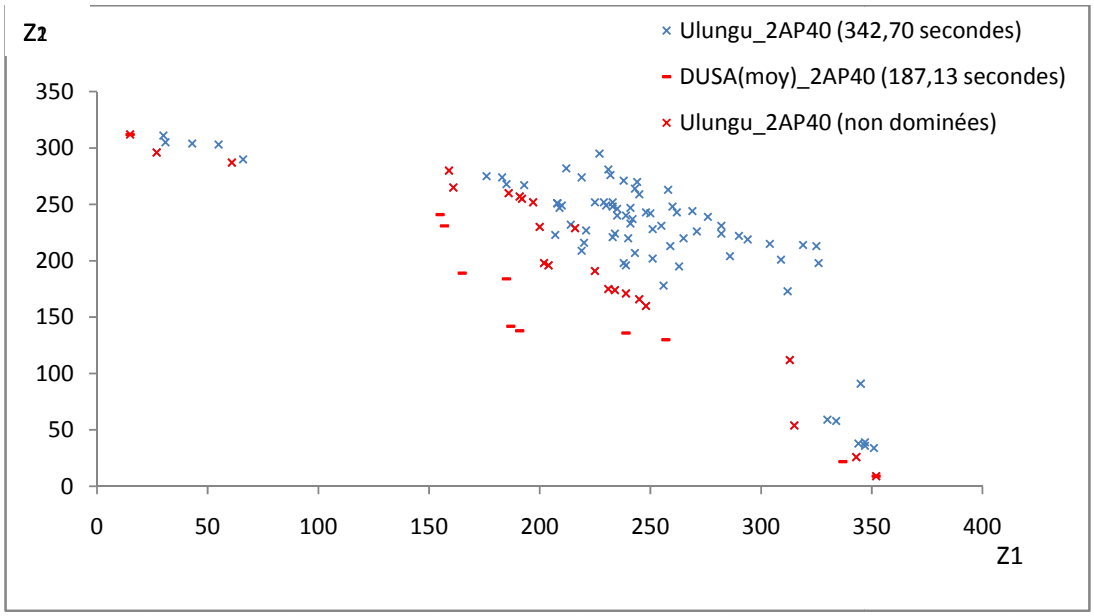


FIG. 3.11: Comparaison de MOSA d'Ulungu et de DUSA(average) pour 2AP40

d'une solution dominée est définie à partir de la plus petite dégradation parmi les dégradations, du vecteur utilité, obtenues suivant les différentes directions de recherche(table 3.1).

– Des résultats plus performants sont obtenus lorsqu'on augmente les paramètres des différentes variantes du recuit simulé utilisant la dominance, sans pour autant dépasser le temps d'exécution du recuit simulé d'Ulungu basé sur l'agrégation et utilisant des paramètres de moindre valeurs (figure 3.11).

3.6 Conclusion

Nous avons présenté, dans ce chapitre, quatre variantes de la méthode du "recuit simulé", pour le contexte multi-objectif basées sur la relation de dominance. Nos approches ont été expérimentées, de manière particulière, sur des instances du problème d'affectation bi-objectif connues dans la littérature. Les résultats obtenus sont de qualités comparables à ceux trouvés par l'algorithme MOSA d'Ulungu, en plus d'améliorer de façon très significative le temps d'exécution qui est, en moyenne, dix fois plus rapide.

Chapitre 4

Une méthode hybride pour la résolution du PAMO

4.1 Introduction

Etant exigeantes en temps de calcul, les méthodes de résolution exactes s'avèrent vite inutilisables dès lors qu'il s'agit de traiter des instances de grande taille. D'autre part, même si par définition, les métaheuristiques réduisent de manière significative l'espace de recherche exploré, la combinatoire associée demeure importante sur des problèmes de taille réelle. Par ailleurs, les méthodes hybrides, combinant les métaheuristiques et/ou les méthodes exactes, contribuent souvent à produire de meilleurs résultats. Dans ce chapitre, une méthode métaheuristique hybride pour le problème d'affectation multi-objectif (avec deux objectifs et plus) est proposée. Cette méthode utilise la méthode du recuit simulé multi-objectif basée sur une approche Pareto [3], et des techniques hybrides de recherche locale [4].

4.2 Approche Hybride

L'idée principale de l'approche hybride proposée consiste à combiner une méthode exacte et une méthode approximative. Plus précisément, elle est basée sur la méthode du recuit simulé multi-objectif utilisant l'approche Pareto DCSA (chapitre 3), avec deux mouvements de recherche locale : recherche locale utilisant une structure de voisinage et recherche locale utilisant la méthode "branch and bound multi-objectif" (chapitre 2 : ici le "branch and bound multi-objectif" est appliqué à un fragment de solution comme un mouvement local).

Cette hybridation est effectuée d'après le mécanisme suivant de génération d'une nouvelle solution à partir d'une solution courante :
Appliquer la méthode du recuit simulé multi-objectif, basée sur l'approche Pareto (DCSA), et pour générer une solution voisine, utiliser :

- La procédure Random_move(x^r,$newx^r$) (décrite ci-après), avec une probabilité $1-p$.
- La procédure Branch_and_bound_move (x^r,$newx^r$) (décrite ci-après), avec une probabilité p.

où les constantes $1-p$ et p sont fixées à 0.9 et 0.1, respectivement. Et représentent la probabilité de générer, à partir de la solution courante, une nouvelle solution par la recherche locale ou la recherche branch and bound multi-objectif, respectivement.

4.2.1 Recherche locale

Dans la recherche locale, nous utilisons une procédure décrivant un mouvement aléatoire. La nouvelle solution est un voisin de la solution courante si et seulement si les deux solutions diffèrent exactement en deux composantes. Il suffit pour cela, de générer à partir de la solution courante (dans l'espace des solutions) deux tâches, ensuite, de permuter entre les ouvriers qui sont effectés à ces tâches (Algorithme 5).

Algorithme 5 Random_move(x^r,$newx^r$)

1: Générer aléatoirement, à partir de x^r, deux indices (tâches) i_1 $(1 \leq i_1 \leq n)$ et i_2 $(1 \leq i_2 \leq n)$;
2: Poser :

$$newx^r(i) = \begin{cases} x^r(i), & \text{si } i \neq i_1, i_2\,; \\ x^r(i_2) & \text{si } i = i_1\,; \\ x^r(i_1) & \text{si } i = i_2\,; \end{cases}$$

4.2.2 Recherche "branch and bound"

L'algorithme "branch and bound", décrit au chapitre deux, est appliqué à l'espace entier de solutions pour déterminer l'ensemble E de toutes les solutions efficaces (supportées et non supportées). Les résultats expérimentaux de cet algorithme ont été présentés dans la table 2.6 du chapitre deux.

Comme il a été observé précédemment, dans le chapitre deux, l'algorithme branch and bound proposé a permi :

– d'éviter l'examen de certaines parties : en effet seulement 44.5% (en moyenne) de nœuds de l'arborescence ont été visités ;
– d'éliminer des branches entières : 61.9% (en moyenne) parmi les nœuds visités ont été stérilisés.

A cause de l'explosion combinatoire, la taille de l'arborescence atteindra rapidement des valeurs astronomiques. Par conséquent, les proportions des nœuds visités et stérilisés ne seront pas significatifs. Pour réduire l'espace de recherche exploré (par la méthode branch and bound) nous proposons une approche hybride, où la méthode branch and bound multi-objectif, présentée au chapitre deux, est utilisée non pas pour explorer l'espace entier des solutions mais plutôt comme un mouvement local appliqué à un fragment de solution pour déterminer à partir de la solution courante, une nouvelle solution non dominée. En effet, à partir d'une solution courante complète de taille n, nous générons aléatoirement un fragment de solution tel que $n-y$ composantes seront fixées. Le nombre y de composantes non fixées est déterminé par l'expression suivante :

$$y = [m \times \ln n] \tag{4.1}$$

Où y, définit le nombre de voisins de la solution courante (complète) en fonction de la taille du problème. La figure 4.1, illustre l'évolution polynomiale de y en fonction de la taille du problème. Cette définition de la structure de voisinage d'une solution courante, va induire une réduction significative de l'espace de recherche exploré, de manière exacte, par la méthode "branch and bound multi objectif" (table 4.1 illustrée par figure 4.2).

$m \times n$	y	Enumération Complète de $y!$	Au lieu de $n!$
2×5	3	6	120
2×10	4	24	3628800
2×15	5	120	1307674368000
2×20	5	120	2432902008176640000
2×25	6	720	15511210043330985984000000
2×30	6	720	2,65252859812191058636308 48e+32
2×40	7	5040	8,15915283247897734345611269596 12e+47

TAB. 4.1: Nombre de voisins

Considérons à une itération r, une solution courante complète, x^r, de taille n. La structure de voisinage adoptée lorsque nous générons de manière exacte une nouvelle

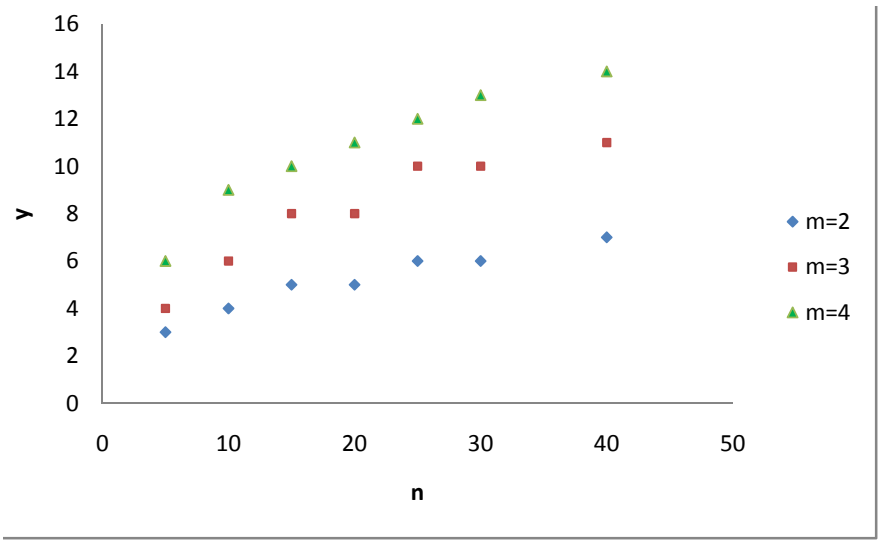


FIG. 4.1: Evolution polynomiale du nombre de composantes non fixées d'un fragment de solution

solution est la suivante : deux solutions sont dites voisines si elles ont exactement $n - y$ composantes communes. Le fragment de x^r représente alors, tous les voisins de x^r. Nous sommes intéressés par la détermination parmi les $y!$ solutions voisines de x^r, celles qui sont non dominées. En d'autres termes, nous sommes intéressés par la détermination, de manière exacte, de toutes les solutions "localement non dominées " dans le voisinage de x^r. La procédure "branch and bound multi objectif" appliquée à l'espace de recherche, définit par le fragment de x^r, est schématisée par Algorithme 6.

Algorithme 6 Branch_and_bound_move(x^r,$newx^r$)

1: Générer, aléatoirement, un fragment de x^r, ayant exactement n-y composantes communes avec celle ci et y autres composantes non fixées.
2: Déterminer par la méthode "branch and bound multi objectif" l'ensemble des solutions "localement non dominées" par x^r.
3: Mettre à jour $\widehat{E}$ par le sous ensemble des voisins de x^r, "localement non dominés ".
4: Choisir aléatoirement, parmi tous les voisins de x^r, "localement non dominés ", une nouvelle solution courante : $newx^r$.

Dans l'exemple de la figure 4.3, la solution x^r, à l'itération r, signifie que le troisième ouvrier est affecté à la tâche T_1, le premier ouvrier est affecté à la tâche T_2, etc .... Pour $n = 5$ et $m = 3$, nous obtenons par l'expression (4.1) que y est égal à 4. Pour obtenir un fragment de la solution courante x^r, nous générons aléatoirement quatre tâches. Pour

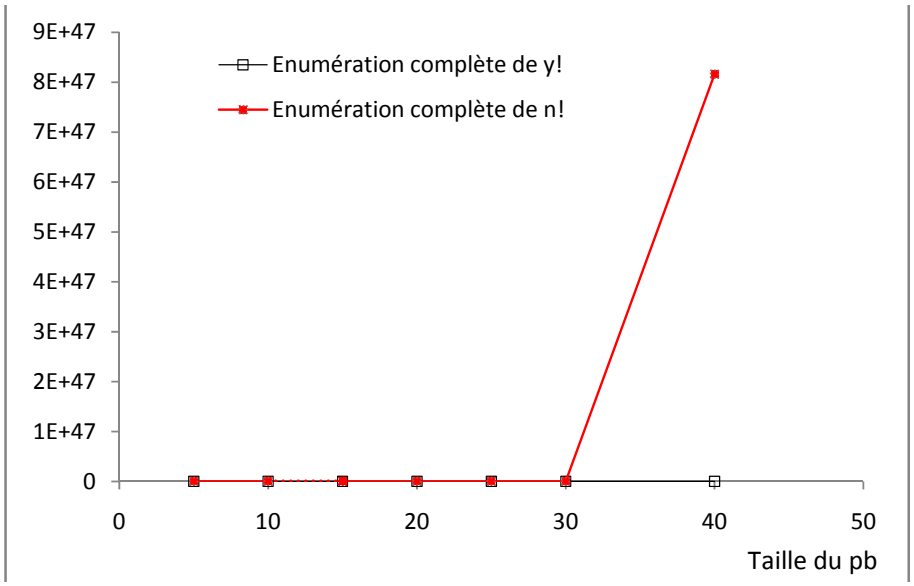


FIG. 4.2: Réduction de l'espace de recherche exploré

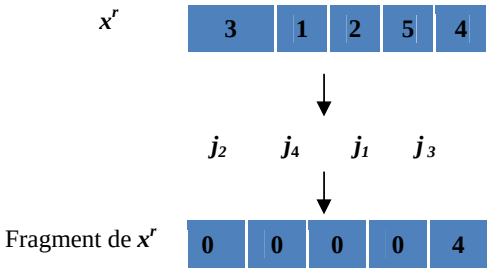


FIG. 4.3: Génération aléatoire d'un fragment d'une solution

cet exemple nous avons généré : T_2, T_4, T_1 et T_3. Le fragment, $(0, 0, 0, 0, 4)$, obtenu de la solution courante $x^r = (3, 1, 2, 5, 4)$, est le sous ensemble de toutes les affectations pour lesquelles l'ouvrier O_4 est affecté à la tâche T_5.

L'exemple suivant illustre comment la méthode branch and bound multi-objectif, est appliquée au fragment de la solution courante représentée dans la figure 4.3, pour déterminer le sous ensemble des solutions "localement non dominées " parmi les $4!$ solutions voisines de $(3, 1, 2, 5, 4)$.

Considérons l'instance suivante du problème d'affectation avec trois objectifs, dont les matrices coût C_1, C_2 et C_3 sont données ci-dessous :

$$\begin{bmatrix} 12 & \underline{5} & 13 & 17 & 19 \\ 12 & 17 & 13 & 10 & 13 \\ \underline{1} & 5 & \underline{9} & \underline{0} & 16 \\ 9 & 8 & 13 & 18 & \underline{9} \\ 12 & 7 & 11 & 0 & 9 \end{bmatrix} \quad \begin{bmatrix} \underline{6} & \underline{3} & 2 & 11 & 17 \\ 6 & 7 & \underline{1} & 5 & 13 \\ 14 & 19 & 7 & 4 & 9 \\ 14 & 10 & 11 & 11 & \underline{2} \\ 8 & 16 & 11 & \underline{3} & 18 \end{bmatrix} \quad \begin{bmatrix} 8 & \underline{5} & 11 & 8 & 5 \\ 6 & 11 & 11 & 2 & 9 \\ \underline{0} & 19 & 17 & 12 & \underline{2} \\ 5 & 12 & \underline{0} & 12 & 2 \\ 16 & 10 & 3 & \underline{1} & 6 \end{bmatrix}$$

Les différentes étapes pour la détermination de l'ensemble, $\{(28, 23, 19); (39, 18, 46); (35, 21, 31); (36, 35, 12)\}$, des solutions "localement non dominées " par (28, 23, 19), dans l'espace des objectifs, sont représentées dans les figures suivantes (de la figure 4.4 à la figure 4.8) :

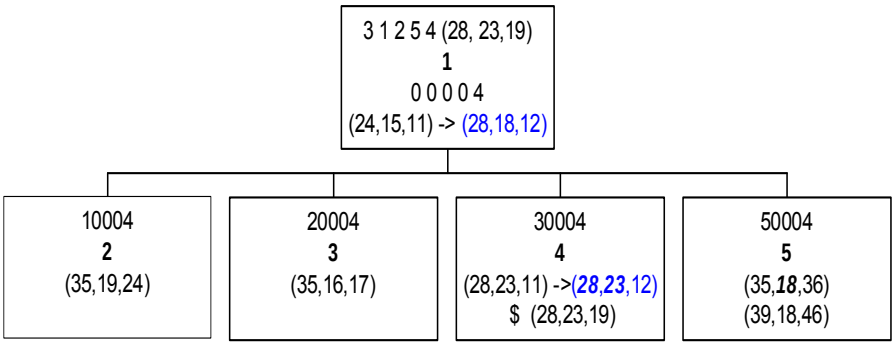


FIG. 4.4: Affectations associées à la tâche T_1 relatives au fragment de la solution donnée par figure 4.3

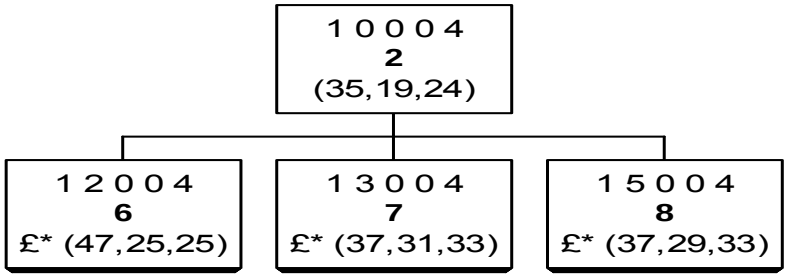


FIG. 4.5: Ouvrier O_1 affecté à la tâche T_1

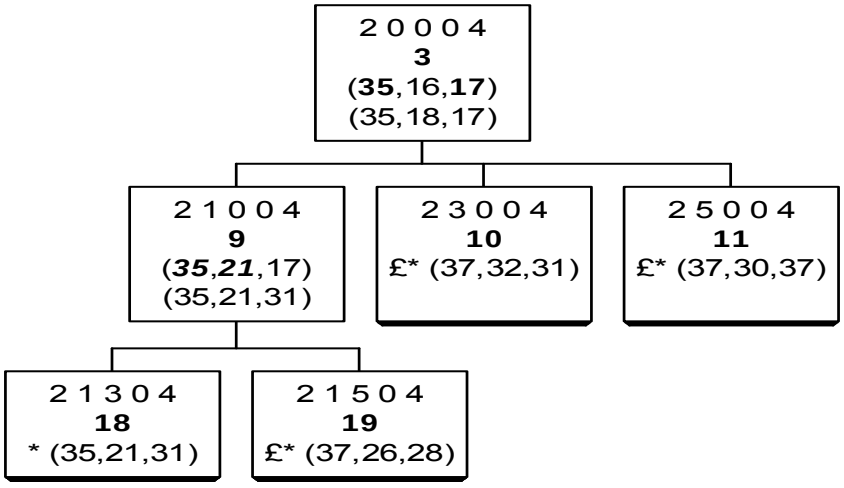


FIG. 4.6: Ouvrier O_2 affecté à la tâche T_1

4.3 Algorithme de la méthode

Une itération de l'algorithme de la méthode est résumée dans Algorithme 7.

Algorithme 7 Une itération de la méthode hybride

1: Générer une nouvelle solution courante $newx^r$ dans le voisinage de x^r
2: De manière aléatoire, avec une probabilité $1 - p$ (procédure Random_move$(x^r, newx^r)$) ;
3: De manière exacte, avec une probabilité p (procédure Branch_and_bound_move$(x^r, newx^r)$)
4: **si** $newx^r$ domine x^r ou aucune des deux solutions x^r et $newx^r$ ne domine l'autre **alors**
5: Remplacer x^r par $newx^r$
6: Mettre à jour $(\widehat{E})$
7: **sinon**
8: Appliquer la règle de Metropolis, basée sur l'approche Pareto, décrite au chapitre trois.
9: **finsi**

L'approche proposée est schématisée dans Algorithme 8, où T_0 est la température initiale, δ est le facteur de décroissance de la température, N_{STOP} est le nombre d'itérations par palier de température et T_{STOP} est la température finale.

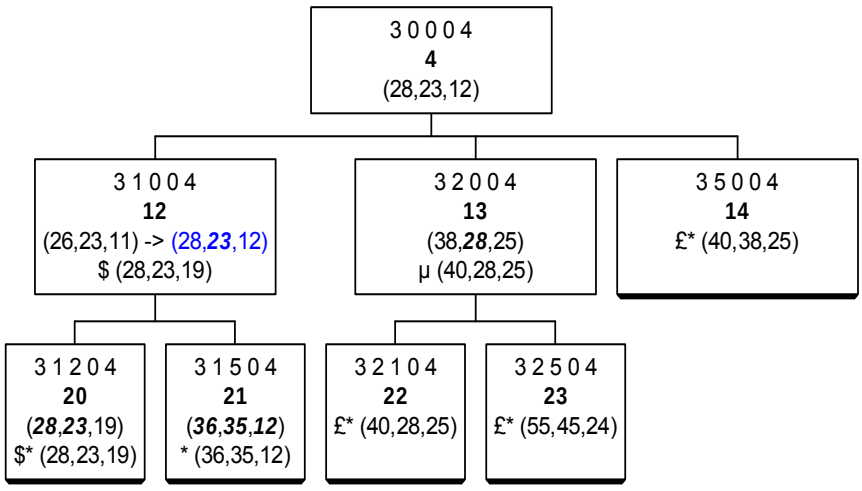


FIG. 4.7: Ouvrier O_3 affecté à la tâche T_1

4.4 Implémentation de la méthode hybride

La méthode hybride proposée a été testée, sur des nouvelles instances du problème d'affectation multi-objectif dont les coefficients des matrices coût sont uniformément générés dans l'intervalle [0,20]. Chaque instance est re-exécutée 10 fois sur un Pentium 4, 1.7 GHz, 256 MB, 20 GB. Elle est comparée à la méthode DCSA d'après les mesures de performances suivantes :

- Proportion de solutions efficaces générées par $\widehat{E}$ (E : ensemble de solutions efficaces) :

$$M_1 = \frac{|\widehat{E} \cap E|}{|E|} \tag{4.2}$$

- La distance moyenne entre les ensembles $\widehat{E}$ et E :

$$D_1(\widehat{E}, E) = \frac{\sum_{x \in E} d(\widehat{E}, x)}{|E|}, \tag{4.3}$$

- La plus mauvaise distance entre $\widehat{E}$ and E :

$$D_2(\widehat{E}, E) = \max_{x \in E} d(\widehat{E}, x), \tag{4.4}$$

- La mesure d'uniformité de l'ensemble $\widehat{E}$:

$$\frac{D_2(\widehat{E}, E)}{D_1(\widehat{E}, E)} \tag{4.5}$$

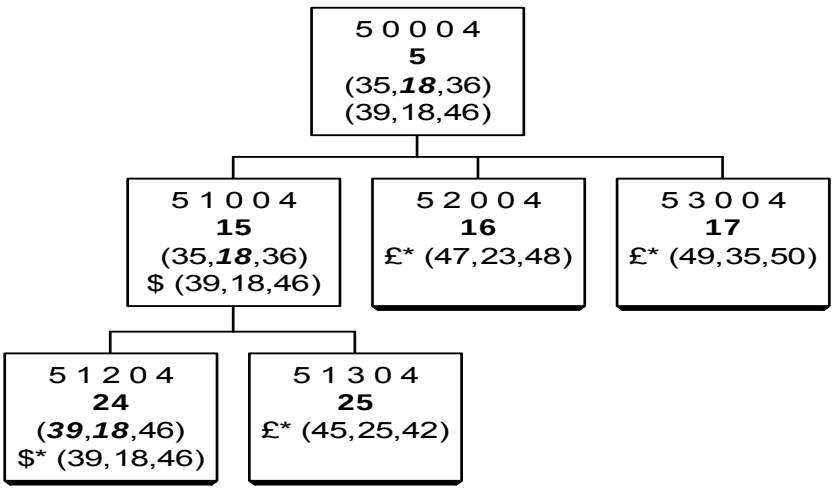


FIG. 4.8: Ouvrier O_5 affecté à la tâche T_1

– L'hypervolume : pourcentage moyen de la frontière efficace [50]

– Le temps d'execution représenté par CPU(t) et mesuré en centième de secondes.

La figure 4.9 met en évidence la forte approximation de HDCSA par rapport à DCSA.

Dans table 4.2 et table 4.3, nous présentons le résumé des résultats obtenus par la méthode DCSA et la méthode hybride HDCSA respectivement sur quelques problèmes d'affectation multi-objectifs. Lorsque nous comparons les deux méthodes, nous pouvons remarquer que, en terme de qualité de solutions, la méthode hybride est plus performante que DMOSA en déterminant plus de 35 % de solutions efficaces supplémentaires pour approximativement le même temps d'exécution. Pour les 6^{me} et 8^{me} instances, la méthode hybride fournie toutes les solutions efficaces.

Algorithme 8 Algorithme de la méthode hybride HDCSA

1: Sélectionner les paramètres de DMOSA (T_0, δ, N_{STOP}, T_{STOP}) ;
2: Générer une solution initiale x_0 ; $\widehat{E} := \{x_0\}$.
3: **répéter**
4: $nbr\_iter := 0$;
5: **répéter**
6: **Algorithme 7**
7: **jusqu'à** $nbr\_iter = N_{STOP}$;
8: $T := \delta \times T$
9: **jusqu'à** $T < T_{STOP}$.
10: $\widehat{E}$ est l'ensemble de solutions potentiellement efficaces.

$m \times n$	$\|E\|$	$\|\widehat{E}\|$	M_1 (%)	D_1	D_2	ratio	hypervolume (%)	Cput time 10^{-2} s
2×5	8	4	30	2.19	7.5	3.43	50	15
2×6	7	3	33	2.86	6.5	2.27	54	17
2×10	16	9	22.3	2.21	7.62	3.45	43.8	18
2×15	39	19	8.1	2.24	7.68	3.42	29.6	20
2×20	54	26	2.5	2.27	7.64	3.36	24	25
3×5	6	3	58	2.89	7	2.42	78	17
3×6	34	7	12	6.57	17	2.59	34.5	22
4×5	32	7	22	7.48	13	1.74	43	16

TAB. 4.2: Performance de la méthode DCSA pour 10000 itérations

4.5 Conclusion

Dans ce chapitre, une approche hybride basée sur la méthode du recuit simulé multi-objectif (DCSA) et utilisant deux mouvements de recherche locale (mouvement aléatoire et mouvement exacte) a été présentée. Elle a été expérimentée, particulièrement, sur le problème d'affectation multi-objectif avec deux objectifs et plus. Les premiers résultats obtenus sont très encourageants. En effet, lorsqu'elle est comparée à DCSA, la méthode hybride HDCSA fournit des solutions qui sont de qualité nettement meilleure pour approximativement la même durée d'exécution.

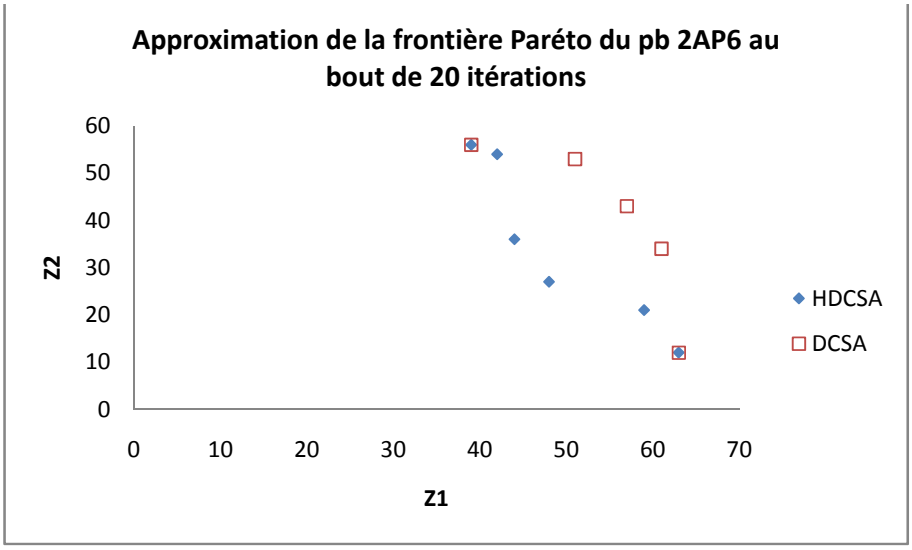


FIG. 4.9: Approximations DCSA et HDCSA

$m \times n$	$\|E\|$	$\|\widehat{E}\|$	M_1 (%)	D_1	D_2	ratio	hypervolume (%)	Cput time 10^{-2} s
2×5	8	7	65	0.69	3	4.35	95	22
2×6	7	6	69	0.64	4.5	7.03	98	22
2×10	16	12	57	0.72	3.2	4.44	87	32
2×15	39	30	44	0.74	3.3	4.46	74	47
2×20	54	43	37.5	0.70	3.1	4.43	56	54
3×5	6	6	100	0	0	-	100	17
3×6	34	18	28	3.13	7	2.24	58	60
4×5	32	32	100	0	0	-	100	22

TAB. 4.3: Performance de la méthode hybride (HDCSA) pour 10000 itérations

Chapitre 5

Comparaison et classement des solutions efficaces

5.1 Introduction

La détermination de toutes les solutions efficaces ne résout pas réellement un problème multi-objectif, car il faudra encore désigner un meilleur compromis parmi celles-ci, qui forment, notamment pour les problèmes de grandes tailles, un ensemble discret mais de cardinal fort élevé. Dans ce chapitre, nous proposons une méthode de surclassement [5] basée sur une relation de dominance flou, sur l'ensemble des directions de recherche, pour la comparaison et le classement des solutions obtenues.

5.2 Construction d'un modèle de préférence

L'idée d'Aide Multicritère à la Décision est traditionnellement présentée dans une logique d'agrégation des objectifs. Dans l'élaboration d'un modèle mathématique, une place importante est accordée à la phase de modélisation des préférences. En effet, cette dernière privilégie la prise en compte d'aspects qualitatifs et de critères (objectifs) multiples difficilement commensurables. Elle est utilisée ensuite dans une procédure d'aide à la décision (optimisation, classement, sélection, etc...).

Soit $\lambda = (\lambda_1, \lambda_2, ..., \lambda_m)^t$ une direction de recherche dont les composantes sont les poids λ_k, $k = 1..m$, associés aux m fonctions objectifs respectivement. On demande aux experts de comparer l'importance des différents objectifs au moyen de relations de préférence que l'on notera (I), (P) et (R) et que nous définissons de la manière suivante

- Z (I) Z' si et seulement si on juge que Z et Z' sont de même importance ;

- Z (P) $Z^{'}$ si et seulement si Z est jugé plus important que $Z^{'}$;
- Z (R) $Z^{'}$ si et seulement si Z et $Z^{'}$ sont incomparables.

Ces relations sont ensuite transformées en contraintes, en associant à chaque objectif son poids λ_k $(k = 1..m)$ possédant les propriétés d'une probabilité, et ceci de la manière suivante :

$$Z^{k_1}(I)Z^{k_2} \Rightarrow |\lambda_{k_1} - \lambda_{k_2}| \leq \delta \tag{5.1}$$

$$Z^{k_1}(P)Z^{k_2} \Rightarrow \lambda_{k_1} - \lambda_{k_2} \geq \delta^{+} \tag{5.2}$$

$$(Z^{k_1} \cup Z^{k_2})(I)(Z^{k_3} \cup Z^{k_4}) \Rightarrow |\lambda_{k_1} + \lambda_{k_2} - \lambda_{k_3} - \lambda_{k_4}| \leq \delta \tag{5.3}$$

$$(Z^{k_1} \cup Z^{k_2})(P)(Z^{k_3} \cup Z^{k_4}) \Rightarrow \lambda_{k_1} + \lambda_{k_2} - \lambda_{k_3} - \lambda_{k_4} \geq \delta^{+} \tag{5.4}$$

et telles que :

$$\lambda_k \geq \delta^{+}; \quad k = 1..m \tag{5.5}$$

autrement dit aucun objectif ne se voit associer un poids nul.

$$\sum_{k=1}^{m} \lambda_k = 1 \tag{5.6}$$

$\delta^{+} = \delta + \epsilon$, ϵ assez petit, δ^{+} et δ sont appelés seuils de préférence. Les contraintes (1)-(6) définissent un domaine de préférence initial $(\wp)$ constituant un polyèdre convexe contenu dans le simplexe de l'espace $\Re^m$ défini par : $\{\lambda \in \Re^m; \sum_{k=1}^{m} \lambda_k = 1, \lambda_k \geq 0, k = 1..m\}$

Remarque 5.1. *Si, pour un certain objectif k, le décideur envisage au préalable de poser $\lambda_k \geq \alpha$, où α est une valeur réelle telle que $\alpha > \delta^{+}$, alors pour éviter une redondance on remplace dans le premier système (système de la première étape) la contrainte $\lambda_k \geq \delta^{+}$ par la contrainte $\lambda_k \geq \alpha$.*

5.3 Réduction progressive du domaine de préférence initial

Afin d'affiner le modèle de représentation des directions de recherche, on définit dans une deuxième étape un processus interactif opérant une réduction progressive du polyèdre

Algorithme 9 Construction du modèle de préférence (des directions de recherche)

ENTRÉES: Modèle de préférence (relations entre différents objectifs :indifférence, préférence et incomparabilité).

SORTIES: Domaine de plausibilité initial (des directions de recherche)

1: Attribuer à chaque objectif Z^k un poids λ_k $(k = 1..m)$ telle que $\sum_{k=1}^{m} \lambda_k = 1$.

2: Définir deux seuils δ^+ et δ (seuils de plausibilité) tels que : $\delta^+ = \delta + \epsilon$, $(\epsilon > 0)$ et $\lambda_k \geq \delta^+$ $(k = 1..m)$.

3: Etablir le système des contraintes définissant le domaine de plausibilité des directions de recherche (polyèdre) de la façon suivante :

- $Z^{k_1}(I)Z^{k_2} \Rightarrow |\lambda_{k_1} - \lambda_{k_2}| \leq \delta$
- $Z^{k_1}(P)Z^{k_2} \Rightarrow \lambda_{k_1} - \lambda_{k_2} \geq \delta^+$
- $(Z^{k_1} \cup Z^{k_2})(I)(Z^{k_3} \cup Z^{k_4}) \Rightarrow |\lambda_{k_1} + \lambda_{k_2} - \lambda_{k_3} - \lambda_{k_4}| \leq \delta$
- $(Z^{k_1} \cup Z^{k_2})(P)(Z^{k_3} \cup Z^{k_4}) \Rightarrow \lambda_{k_1} + \lambda_{k_2} - \lambda_{k_3} - \lambda_{k_4} \geq \delta^+$

de départ $(\wp)$ et ceci par l'adjonction de nouvelles contraintes sur les poids λ_1, λ_2, ..., λ_m associés aux objectifs Z^1, Z^2, ..., Z^m. Il s'agit de faire réagir les experts sur quelques directions de recherche "limites" exhibées à partir de l'exploration des frontières de $(\wp)$, à savoir :

$$(P_k^{max}) \begin{cases} max\ \lambda_k \\ \lambda \in \wp \end{cases}$$

$$(P_k^{min}) \begin{cases} min\ \lambda_k \\ \lambda \in \wp \end{cases}$$

où $\lambda = (\lambda_1, \lambda_2, ..., \lambda_m)^t$.

En leur posant les questions suivantes :

- jugez-vous que cette direction de recherche est cohérente et réaliste ?
- si elle ne l'est pas, quelle est la contrainte sur les λ_k qui n'est pas satisfaite par cette direction et par conséquent qu'il faudrait rajouter aux contraintes initiales ?

5.4 Discrétisation du domaine de préférence final

Le polyèdre $(\wp_f)$ construit à partir de la méthode interactive citée ci-dessus (algorithme 10), regroupe toutes les directions de recherches "réalisables" compte tenu de l'information disponible sur les préférences respectives des objectifs considérés. Ainsi, pour

Algorithme 10 Réduction progressive du domaine de préférence

Entrées: Domaine de préférence initial (des directions de recherche).

Sorties: Domaine de préférence final (des directions de recherche).

1: Résoudre pour chaque k ($k = 1..m$), les problèmes suivants :

$$(P_k^{max}) \begin{cases} max\ \lambda_k \\ \lambda \in \wp \end{cases}$$

$$(P_k^{min}) \begin{cases} min\ \lambda_k \\ \lambda \in \wp \end{cases}$$

2: Classer les problèmes : les m premiers sont du type (P_k^{max}) et les m derniers sont du type (P_k^{min}).

3: Juger de la cohérence des différentes directions de recherche limites qui représentent les solutions des $2m$ problèmes cités ci-dessus :

4: Poser $I = \{1, 2, ..., m, m+1, ..., 2m\}$

5: k :=1 ;

6: **répéter**

7: **If** λ_k est cohérente **Then** supprimer k de I ($I := I \setminus \{k\}$).

8: **Else** ajouter la contrainte supplémentaire violée.

9: Soit $(wp)'$ le nouveau polyèdre obtenu, poser $(wp) := (wp)'$.

10: $k := k + 1$.

11: **jusqu'à** $k = |I|$

12: **If** $I = \emptyset$ **Then** toutes les directions de recherche sont cohérentes d'où le polyèdre de préférence est final.

13: **Else** Itérer l'étape 1

une solution donnée, on peut faire correspondre à chaque point du polyèdre $(\wp_f)$ (ou direction de recherche) une utilité (fonction d'agrégation) particulière. Afin de rendre plus facile la comparaison entre solutions, il est commode d'associer à chacune de ces solutions un nombre fini d'utilités, et par conséquent de discrétiser le domaine de préférence $(\wp_f)$ (algorithme 11). La procédure de discrétisation consiste à calculer un nombre fini de combinaisons convexes des sommets du polyèdre final $(\wp_f)$. Comme $(\wp_f)$ est borné alors tout point y appartenant à l'interieur de $(\wp_f)$ s'écrit comme combinaison convexe des sommets de $(\wp_f)$, c'est à dire $y = \sum_{i=1}^{q} \alpha_i x^i$ où $x^1, x^2, ..., x^q$ sont les q sommets de $(\wp_f)$, avec $\sum_{i=1}^{q} \alpha_i = 1, \alpha_i \geq 0$. Afin que les différentes combinaisons convexes ainsi obtenues soient bien dispersées, il suffit que les coefficients α_i $(i = 1..q)$, soient différents les uns des autres et par conséquent, soient bien dispersés aussi. Leur dispersion dépend essentiellement de la distribution utilisée pour calculer ces derniers.

5.4.1 Combinaisons convexes suivant la loi uniforme

Soient $x^1, x^2, ..., x^q$ les sommets du polyèdre $(\wp_f)$.

- générer aléatoirement des valeurs $\beta_1, \beta_2, ..., \beta_q$ selon la loi uniforme (i.e. $0 < \beta_j < 1; j = 1..q$)
- poser $\alpha_i = \frac{\beta_i}{\sum_{j=1}^{q} \beta_j}$ ce qui donne $\sum_{i=1}^{q} \alpha_i = 1$
- calculer $y = \sum_{i=1}^{q} \alpha_i x^i$.

On obtient de cette manière, un certain nombre de directions de recherche $y \in (\wp_f)$ concentrées au centre du polyèdre [76].

5.4.2 Combinaisons convexes suivant la loi de Weibull

Soient $x^1, x^2, ..., x^q$ les sommets du polyèdre $(\wp_f)$.

- générer aléatoirement des valeurs $\beta_1, \beta_2, ..., \beta_q$ selon la loi de Weibull, de paramètre b et c donneé par l'expression suivante : $\beta_i = b[-\ln(\delta_i)]^{\frac{1}{c}}$; $i = 1..q$ où δ_i est généré selon la loi uniforme sur $[0, 1]$.
- poser $\alpha_i = \frac{\beta_i}{\sum_{j=1}^{q} \beta_j}$ $i = 1..q$
- calculer $y = \sum_{i=1}^{q} \alpha_i x^i$.

Si on considère les valeurs 0.1 et 0.3 pour les paramètres b et c respectivement, alors la répartition géométrique des combinaisons convexes est dans ce cas orientée vers les extrémités du polyèdre [76].

A la fin de cette opération , on aboutit à $10 \times J$ directions de recherche $\lambda^j = (\lambda_1^j, \lambda_2^j, ..., \lambda_m^j)$, $j = 1..10 \times J$.

Algorithme 11 Discrétisation du domaine de préférence final

ENTRÉES: Domaine de préférence final continu ($\wp$).

SORTIES: Domaine de préférence final discrétisé.

1: Trouver tous les sommets de ($\wp$) (calculer toutes les solutions de base du programme linéaire définit par ($\wp$)).
2: Déterminer de façon aléatoire un certain nombre de points $y \in (\wp)$:
   – générer aléatoirement 50% des combinaisons convexes suivant la loi uniforme, et 50% d'autres suivant la loi de Weibull de paramètres $b = 0.1$ et $c = 0.3$.

5.5 Dispersion des directions de recherche

Pour assurer une bonne dispersion des directions de recherche générées par la procédure de discrétisation, nous opérons un filtrage sur l'ensemble des $10 \times J$ directions de recherches pour obtenir J directions de recherches qui étant associées à une norme donnée, sont les plus éloignées les unes des autres.

5.5.1 Mesure de distance (norme L_p)

Soit $\Lambda \subset \Re^m$ un ensemble fini de vecteurs. Le processus de filtrage s'opère en comparant les distances entre les paires de vecteurs $\lambda^t, \lambda^h \in \Lambda$. La distance entre λ^t et λ^h, associée à la norme métrique L_p est donnée par :

$$\|\lambda^t - \lambda^h\|_p^\pi = [\sum_{k=1}^{m} (\pi_k |\lambda_k^t - \lambda_k^h|)^p]^{\frac{1}{p}}, \tag{5.7}$$

où :

– m est la dimension des vecteurs à filtrer ;
– p est le paramètre métrique de la norme L_p, $p \in (\{1, 2, ...\} \cup \{\infty\})$.
– π_k est l'indice de normalisation associé à la k^{eme} composante, des différents vecteurs à filtrer. Il est calculé par l'expression :

$$\pi_k = \frac{1}{R_k} [\sum_{j=1}^{m} \frac{1}{R_j}]^{-1}, \tag{5.8}$$

tel que :

$$R_k = \max_{\lambda \in \Lambda} \{\lambda_k\} - \min_{\lambda \in \Lambda} \{\lambda_k\}. \tag{5.9}$$

5.5.2 Mécanisme de filtrage

Il existe dans la littérature plusieurs méthodes de filtrage [76]. Nous citons parmi ces dernières, la méthode **"du premier point à l'extérieur du voisinage "**, **"le plus loin point à l'extérieur du voisinage "**, ... etc. Dans toutes ces méthodes on utilise la relation du filtrage suivante :

$$[\sum_{k=1}^{m}(\pi_k|\lambda_k^t - \lambda_k^h|)^p]^{\frac{1}{p}} < d, \tag{5.10}$$

pour comparer la distance entre directions de recherche. Le paramètre, d, représente le test de distance. Il permet de gérer le processus du filtrage dans le sens suivant :

- Si la distance entre λ^t et λ^h est supérieure ou égale à d, alors les deux directions de recherche sont supposées être significativement différentes l'une de l'autre. Par contre :
- Si la distance entre λ^t et λ^h est inférieur à d, alors les deux directions de recherche sont supposées être significativement différentes l'une de l'autre.

Parmi les différentes méthodes existantes, notre choix s'est orienté vers la méthode **"du premier point à l'extérieur du voisinage "**. Notons cependant qu'il existe d'autres méthodes donnant des résultats plus performants mais qui sont plus lentes et plus compliquées à implémenter.

5.5.3 Méthode "du premier point à l'extèrieur du voisinage"

Supposons que nous avons $10 \times J$ directions de recherche et que nous souhaitons obtenir un ensemble dispersé de taille J, en utilisant la méthode "du premier point à l'extérieur du voisinage". Initialement on commence par fixer la valeur d, du paramètre test de distance. Ensuite, on place les $10 \times J$ directions de recherche à filter dans une liste.

La première direction de recherche retenue par le filtrage, sera le premier élément de la liste des $10 \times J$ directions de recherche. La seconde direction de recherche retenue par le filtrage, sera une direction de recherche parmi les $((10 \times J) - 1)$ éléments restant de la liste, et qui soit significativement différente de la première direction de recherche retenue par filtrage. La troisième direction de recherche retenue par le filtrage, sera une direction de recherche parmi les $((10 \times J) - 2)$ éléments restant de la liste, et qui soit significativement simultanément différente de la première et de seconde directions de recherche retenues par le filtrage. La j^{me} direction de recherche retenue par le filtrage, sera une direction de recherche parmi les $(10 \times J) - j + 1$ éléments de la liste, et qui soit significativement

simultanément différente de la 1^{re}, 2^{me}, ...,$(j-1)^{me}$ directions de recherche retenues par le filtrage. Le processus de filtrage s'arrêtre lorsque aucune des directions de recherche restantes dans la liste ne vérifie cette condition.

Notons que l'on peut toutes fois achever le processus du filtrage sans pour autant parvenir à la taille souhaitée de l'ensemble dispersé, le nombre de directions de recherche filtrés est dans ce cas soit supérieur à J soit inférieur à J. La raison pour laquelle cette possibilité peut survenir, est liée à la valeur fixée du paramètre test de distance qui est dans ce cas soit trop petite soit trop grande. Malheureusement, en dehors des méthodes expérimentales [76], il n'existe pas d'autres qui déterminent la valeur approchée du paramètre test de distance.

A la fin de l'opération de filtrage, on aboutit à J directions de recherches. Le vecteur $(U^1(x), U^2(x), ..., U^J(x))$, où $U^j(x) = U(x, \lambda^j) = \sum_{k=1}^{m} \lambda_k^j . C^k(x)$, $j = 1..J$, représente les utilités multiples de la solution x suivant les différentes directions de recherches $\lambda^1, \lambda^2, ..., \lambda^J$.

En utilisant ces directions de recherche, nous pouvons déterminer à l'aide des méthodes du recuit simulé multi-objectif proposées au chapitre trois ($DUSA(\theta)$), un ensemble de solutions potentiellement efficaces ($\widehat{E}$) qui représente une bonne approximation de l'ensemble de solutions efficaces (E).

5.6 Comparaison des solutions potentiellement efficaces

Soit $\widehat{E} = \{a_1, a_2, ..., a_{nb}\}$ l'ensemble des actions à comparer.

On se propose d'explorer le domaine de préférence discrétisé (des directions de recherches) en recherchant parmi les solutions potentiellement efficaces (mutuellement non dominées) un sous-ensemble flou des actions non dominées la solution "la moins dominée". On définit sur l'ensemble $\widehat{E} \times \widehat{E}$, une relation floue de surclassement S_f, dont la fonction d'appartenance μ_{S_f} mesure la valeur de vérité de l'assertion : "a_i est jugée au moins aussi bonne que a_h" de la manière suivante :

$$\mu_{S_f}(a_i, a_h) = \begin{cases} \frac{|\{j, U^j(a_i) \leq U^j(a_h)\}|}{J}, & \text{si } \nexists j^0 \text{ tel que } U^{j^0}(a_i) + v > U^{j^0}(a_h)\,; \\ 0 & \text{sinon,} \end{cases}$$

où v est un seuil de veto (par exemple : v = 0.2). $\mu_{S_f}(a_i, a_h)$ est par conséquent égale à la proportion d'utilités pour lesquelles a_h n'est pas préférée à a_i.

Si pour une direction de recherche j^0, l'écart entre $U^{j^0}(a_i)$ et $U^{j^0}(a_h)$ est par trop défavorable à a_i, alors on refuse toute crédibilité au surclassement de a_h par a_i quelles que soient les performances de ces deux solutions pour les autres directions de recherches. La relation S_f permet ainsi de modéliser des situations de :

- préférence stricte : $\mu_{S_f}(a_i, a_h) = 1$ et $\mu_{S_f}(a_h, a_i) = 0$,
- indifférence : $\mu_{S_f}(a_i, a_h) = \mu_{S_f}(a_h, a_i) = 1$,
- incomparabilité : $\mu_{S_f}(a_i, a_h) = \mu_{S_f}(a_h, a_i) = 0$,

ainsi que toutes les situations intermédiaires où une solution n'est meilleure qu'une autre que sur une partie du domaine de préférence.

A partir de la relation de surclassement S_f, on définit sur $\widehat{E} \times \widehat{E}$ la relation floue de dominance (ou de préférence) D, dont la fonction d'appartenance est définie comme suit :

$$\mu_D(a_i, a_h) = \begin{cases} \mu_{S_f}(a_i, a_h) - \mu_{S_f}(a_h, a_i), & \text{si } \mu_{S_f}(a_i, a_h) > \mu_{S_f}(a_h, a_i)\,; \\ 0, & \text{sinon.} \end{cases}$$

Pour une valeur de a_i fixée, $\mu_D(a_i, a_h)$ représente le sous-ensemble flou des actions a_h dominées par a_i. Son complémentaire, défini par la fonction d'appartenance $1 - \mu_D(a_i, a_h)$ est le sous-ensemble flou des actions non dominées par a_i. L'intersection de tous les sous-ensembles flous des actions non dominées par a_i, lorsque a_i parcourt $\widehat{E}$ donne le sous-ensemble des actions qui ne sont dominées par aucune autre et dont la fonction d'appartenance est définie par :

$$\mu^{ND}(a_h) = \inf[1 - \mu_D(a_i, a_h), a_i \in \widehat{E}] = 1 - \sup[\mu_D(a_i, a_h), a_i \in \widehat{E}]. \tag{5.11}$$

Comme

$$\sup[\mu_D(a_i, a_h) = \sup[\mu_{S_f}(a_i, a_h) - \mu_{S_f}(a_h, a_i)], \tag{5.12}$$

alors on obtient :

$$\mu^{ND}(a_h) = 1 - \sup[\mu_{S_f}(a_i, a_h) - \mu_{S_f}(a_h, a_i)]. \tag{5.13}$$

$\mu^{ND}(a_h)$ peut être interprétée comme le degré de vérité de l'assertion : " a_h n'est dominée par aucune action de $\widehat{E}$ ".

Lorsque l'on recherche la meilleure ou les meilleures actions (problématique d'optimisation), il est donc logique de choisir celle dont la valeur de μ^{ND} est la plus proche de 1. Si on souhaite obtenir un classement complet des actions, il est nécessaire de procéder par étapes successives en éliminant les actions déjà classées et en recalculant μ^{ND} à chaque fois.

Algorithme 12 Choix parmi les solutions potentiellement efficaces

ENTRÉES: Modèle de préférence (relations entre différents objectifs :indifférence, préférence et incomparabilité).

SORTIES: Classement complet des solutions de $\widehat{E}$.

1: Construction du domaine de préférence initial (pour les directions de recherche) ;
2: Réduction progressive du domaine de préférence (procédure interactive) ;
3: Discrétisation du domaine final et génération de J directions de recherche
4: Approximation de l'ensemble de solutions efficaces, E, par la méthode $DUSA(\theta)$ (voir chapitre 3). Soit $\widehat{E}$, cette approximation.
5: Recherche du sous ensemble flou des actions $a_i \in \widehat{E}$ non dominées.

5.7 Application à un exemple didactique

Considérons l'instance (3AP5) suivante du problème d'affectation avec trois objectifs, dont les matrices coût C_1, C_2 et C_3 sont données ci-dessous :

$$\begin{bmatrix} 12 & \underline{5} & 13 & 17 & 19 \\ 12 & 17 & 13 & 10 & 13 \\ \underline{1} & 5 & \underline{9} & \underline{0} & 16 \\ 9 & 8 & 13 & 18 & \underline{9} \\ 12 & 7 & 11 & 0 & 9 \end{bmatrix} \quad \begin{bmatrix} \underline{6} & \underline{3} & 2 & 11 & 17 \\ 6 & 7 & \underline{1} & 5 & 13 \\ 14 & 19 & 7 & 4 & 9 \\ 14 & 10 & 11 & 11 & \underline{2} \\ 8 & 16 & 11 & \underline{3} & 18 \end{bmatrix} \quad \begin{bmatrix} 8 & \underline{5} & 11 & 8 & 5 \\ 6 & 11 & 11 & 2 & 9 \\ \underline{0} & 19 & 17 & 12 & \underline{2} \\ 5 & 12 & \underline{0} & 12 & 2 \\ 16 & 10 & 3 & \underline{1} & 6 \end{bmatrix}$$

5.7.1 Construction du modèle de préférence

Soient λ_1, λ_2 et λ_3 les poids associés aux objectifs Z^1, Z^2 et Z^3 respectivement. On a alors : $\sum_{k=1}^{3} \lambda_k = 1$. Et pour un seuil $\delta^+ = 0.1$, on a : $\lambda_k \geq 0.1$ $k = 1..3$

Il existe entre les objectifs la relation de préférence (P) et la relation d'indifférence (I). On a pour $\delta = 0.05$ le système suivant :

$$\begin{cases} Z^2(P)Z^1 & \Rightarrow & \lambda_2 - \lambda_1 \geq 0.1 \\ Z^2(P)Z^3 & \Rightarrow & \lambda_2 - \lambda_3 \geq 0.1 \\ Z^1(I)Z^3 & \Rightarrow & |\lambda_1 - \lambda_3| \leq 0.05 \end{cases}$$

Le domaine de préférence des directions de recherche (le polyèdre $(\wp)$) est donné par le système de contraintes suivant :

$$\begin{cases} -\lambda_1 + \lambda_2 \geq 0.1 & (1) \\ \lambda_2 - \lambda_3 \geq 0.1 & (2) \\ \lambda_1 - \lambda_3 \leq 0.05 & (3) \\ -\lambda_1 + \lambda_3 \leq 0.05 & (4) \\ \lambda_1 \geq 0.1 & (5) \\ \lambda_2 \geq 0.1 & (6) \\ \lambda_3 \geq 0.1 & (7) \\ \lambda_1 + \lambda_2 + \lambda_3 = 1 & (8) \end{cases}$$

Etant donné qu'il a été décidé, à l'avance, que $\lambda_2 \geq 0.5$, alors on remplace la contrainte (6) par la contrainte : $\lambda_2 \geq 0.5$ On obtient alors le système :

$$\begin{cases} -\lambda_1 + \lambda_2 \geq 0.1 & (1) \\ \lambda_2 - \lambda_3 \geq 0.1 & (2) \\ \lambda_1 - \lambda_3 \leq 0.05 & (3) \\ -\lambda_1 + \lambda_3 \leq 0.05 & (4) \\ \lambda_1 \geq 0.1 & (5) \\ \lambda_2 \geq 0.5 & (6) \\ \lambda_3 \geq 0.1 & (7) \\ \lambda_1 + \lambda_2 + \lambda_3 = 1 & (8) \end{cases}$$

5.7.2 Réduction progressive du domaine de préférence

Les résultats de la génération de quelques points extrémaux de ($\wp$) (directions de recherche limites) ainsi que leur cohérence et les contraintes rajoutées dans le cas de non cohérence sont donnés dans la Table 5.1 On recommence avec le nouvel ensemble de contraintes pour trouver les résultats présentés dans la Table 5.2

L'ensemble définitif des contraintes s'établit donc ainsi :

Objectif	Direction de recherche générée	Cette direction de recherche	Quelle contrainte
	$\lambda_1; \lambda_2; \lambda_3$	est-elle jugée cohérente ?	faut-il ajout ?
$Max\lambda_1$	0.275 ; 0.500 ; 0.225	oui	–
$Max\lambda_2$	0.100 ; 0.800 ; 0.100	oui	–
$Max\lambda_3$	0.225 ; 0.500 ; 0.275	non	$\lambda_3 \leq 0.25$
$Min\lambda_1$	0.100 ; 0.800 ; 0.100	oui	–
$Min\lambda_2$	0.275 ; 0.500 ; 0.225	oui	–
$Min\lambda_3$	0.150 ; 0.750 ; 0.100	oui	–

TAB. 5.1: Les points extrémaux du polyèdre ($\wp$) et leur cohérence

Objectif	Direction de recherche générée	Cette direction de recherche	Quelle contrainte
	λ_1 ; λ_2 ; λ_3	est-elle jugée cohérente ?	faut-il ajout ?
$Max\lambda_3$	0.200 ; 0.550 ; 0.250	oui	–

TAB. 5.2: Les points extrémaux du polyèdre correspondant au nouvel ensemble des contraintes et leur cohérence

$$\begin{cases} -\lambda_1 + \lambda_2 \geq 0.1 \\ \lambda_2 - \lambda_3 \geq 0.1 \\ \lambda_1 - \lambda_3 \leq 0.05 \\ -\lambda_1 + \lambda_3 \leq 0.05 \\ \lambda_1 \geq 0.1 \\ \lambda_2 \geq 0.5 \\ \lambda_3 \geq 0.1 \\ \lambda_3 \leq 0.25 \\ \lambda_1 + \lambda_2 + \lambda_3 = 1 \end{cases}$$

Le système ($\wp_f$) définit le polyèdre final qu'on va discrétiser.

5.7.3 Discrétisation du domaine de préférence final

La discrétisation du polyèdre final ($\wp_f$) est réalisé à partir de la détermination de toutes les solutions de base réalisables de ce dernier. En effet, comme $\sum_{k=1}^{3} \lambda_k = 1$ alors le polyèdre ($\wp_f$) est borné. Par conséquent chaque point de ce polyèdre (ou direction de recherche) s'écrit comme combinaison convexe des cinq sommets de ($\wp_f$) notés P^1, P^2,

P^3, P^4 et P^5 et qui sont donnés dans la Table 5.3

Sommets de $\wp_f$	λ_1	λ_2	λ_3
P^1	0.200	0.550	0.250
P^2	0.250	0.500	0.250
P^3	0.100	0.750	0.150
P^4	0.270	0.500	0.230
P^5	0.100	0.800	0.100

TAB. 5.3: Solutions de base réalisables (sommets) du polyèdre final ($\wp_f$)

En limitant le nombre de jeux de poids de la discrétisation à 10, nous retrouvons les résultats donnés dans la Table 5.4

Indice	Jeu de poids de la discrétisation
j	$\lambda_1; \lambda_2; \lambda_3$
1	0.1604 ; 0.6614 ; 0.1782
2	0.2265 ; 0.5572 ; 0.2163
3	0.1985 ; 0.6068 ; 0.1947
4	0.1637 ; 0.6458 ; 0.1906
5	0.1765 ; 0.6443 ; 0.1793
6	0.1855 ; 0.6196 ; 0.1949
7	0.1822 ; 0.6216 ; 0.1922
8	0.1858 ; 0.6286 ; 0.1856
9	0.1553 ; 0.6737 ; 0.1711
6	0.1540 ; 0.6635 ; 0.1825

TAB. 5.4: Les dix jeux de points de la discrétisation

Les résultats de l'approximation, dans l'espace des objectifs, de l'ensemble de solutions efficaces, par la méthode $DUSA(\theta)$, sont donnés dans la Table 5.5

Le classement des six solutions potentiellement efficaces générées par la méthode $DUSA(\theta)$ est obtenu après le calcul des valeurs $\mu^{ND}(a_i$ pour $i = 1..6$ et pour un seuil de veto $v = 0.2$. Le résultat de ce classement est présenté dans la Table 5.6

Les vecteurs coût non dominés ayant des faibles valeurs au niveau de la deuxième composante, correspondant au second objectif, se voient occupés les premiers rangs. D'autre

N^0	vecteur coût non dominé
1	$(28^*, 23, 19)$
2	$(39, 18^*, 46)$
3	$(36, 35, 12^*)$
4	$(35, 21, 31)$
5	$(46, 32, 14)$
6	$(32, 44, 15)$

TAB. 5.5: Approximation de 3AP5 par la méthode $DUSA(\theta)$

Rang	vecteur coût non dominé
1	$(28^*, 23, 19)$
2	$(35, 21, 31)$
3	$(39, 18^*, 46)$
4	$(36, 35, 12^*)$
5	$(46, 32, 14)$
6	$(32, 44, 15)$

TAB. 5.6: Classement des solutions de 3AP5 générées par la méthode $DUSA(\theta)$

part, on constate que parmi ces mêmes vecteurs, une priorité est accordée à ceux dont le premier et le troisième objectifs s'approchent le plus possible de leur valeurs respectives à l'optimum. Ce ci est justifié par le fait que le premier objectif est indifférent au troisième objectif et vice verca.

5.7.4 Conclusion

Dans ce chapitre avons présenté une méthode de surclassement (multicritère) définie sur l'ensemble de solutions efficaces obtenues de manière exacte ou approché. Cette méthode est basée sur une relation de dominance floue définie sur un ensemble de directions de recherche établi à partir d'un modèle de préférence et qui reflète l'importance accordée aux différents objectifs .

Conclusion

Dans cette thèse, nous nous sommes intéressés aux différentes méthodologies de résolution du problème d'affectation multiobjectif . Plusieurs contributions ont été apportées sous forme de conception, de développement et d'implémentation de plusieurs algorithmes d'optimisation différents. D'abord, nous avons proposé une adaptation de la méthode branch and bound au contexte multi objectif du problème d'affectation (MOBB). Nous avons présenté aussi, plusieurs définitions liées aux principes d'évaluation et de stérilisation dans un contexte multi objectif, d'une méthode branch and bound. L'intérêt majeur, de cette première contribution, est de permettre la résolution des problèmes d'affectations multi objectifs avec trois objectifs et plus contrairement à toutes les méthodes (exactes) existantes qui sont limitées à la résolutions des problèmes d'affectation bi objectif. De plus la méthode branch and bound proposée, conserve le contexte naturel multi objectif du problème sans changer sa structure initiale. Dans les expérimentations réalisées, nous avons pu observer la souhaitabilité d'application des techniques métaheuristiques pour ce problème. En effet, le nombre de solutions efficaces croît très vite avec la taille de l'instance à traiter (explosion combinatoire), ce qui engendre des temps de calcul rapidement prohibitifs (dépassant par exemple l'espérance de vie du système solaire).

L'application de la métaheuristique du recuit simulé pour le problème PAMO, a été introduite pour la première fois par Ulungu [88]. C'est un algorithme stochastique qui permet de calculer des solutions efficaces approchées du front Pareto. Deux nouvelles variantes de l'adaptation de la méthode du recuit simulé au contexte multi objectif sont proposées (DUSA et DCSA). Contrairement à l'approche traditionnelle d' Ulungu (approximation en aval), nos approches consistent à agir en amont. La stratégie d'acceptation ou de rejet d'une solution dominée est définie à partir d'une relation de dominance utilisant toutes les directions de recherche en même temps. Ces variantes sont décrites avec précision dans le cas du problème d'affectation multi-objectif et leurs performances sont illustrées sur des problèmes test standard tirés de la littérature (cas bi-objectif). D'autre

part, même si par définition, les métaheuristiques réduisent de manière significative l'espace de recherche exploré, la combinatoire associée demeure importante sur des problèmes de taille réelle.

En continuant nos travaux, nous avons développé une méthode hybride, HDCSA, qui améliore DCSA sur un aspect essentiel : la phase d'intensification. Ainsi, nous avons développé un processus d'intensification utilisant deux mouvements de recherche locale : mouvement aléatoire et mouvement exact. Les expérimentations menées sur un nouvel ensemble de problèmes tests à deux, à trois et à quatre objectifs, ont permis de mettre en évidence des résultats plus performants par rapport à l'algorithme de référence DCSA.

Notre dernière contribution a trait à l'analyse multicritère. Elle consiste à concevoir, de manière interactive, un modèle de préférence pour la détermination d'un ensemble de directions de recherche à utiliser et la mise au point d'une approximation partielle de l'ensemble des solutions efficaces. La construction d'une relation de dominance floue, utilisant ces directions de recherche, permet alors de classer les solutions obtenues suivant un ordre de préférence du décideur.

A la lumière de ce travail, plusieurs perspectives se dégagent :

- Les premières perspectives envisagées concernent les améliorations de nos algorithmes, par leur transcription dans un environnement Matlab. Ceci va permettre d'une part, de générer de nouvelles instances du problème d'affectation multi-objectif suivant d'autres lois pour étudier l'incidence de la corrélation, entre les matrices "coût", sur la qualité des approximations générées. Et d'autre part, de représenter en dimension trois les frontières Pareto déterminées par la méthode branch and bound pour dégager, eventuellement, des caractérisations géoméoriques des solutions non supportées du problème d'affectation avec trois objectifs.
- La deuxième perspective, concerne l'évaluation des nouvelles variantes proposées, de l'adaptation de la métaheuristique du recuit simulé au contexte multi objectif, sur d'autres problèmes d'optimisation combinatoire multi-objectif : problème du sac à dos multi-objectif, problème du voyageur de commerce, ...
- La troisième perspective, concerne le développement de versions parallèles de la méthode hybride HDCSA, au niveau du processus d'intensification de la solution courante, suivant tous ses voisins "localement non dominés ".
- La quatrième perspective, concerne la définition d'une relation flou de dominance à l'interieur même du processus du recuit simulé, pour déterminer une approximation

partielle qui représente un sous ensemble flou de solutions potentiellement non dominées. Des travaux, dans ce sens, sont en cours d'experimentations pour le problème du sac à dos multi-objectif....

– Enfin, intégrer les outils d'estimation de la qualité du front Pareto pour évaluer les performances qualitatives des méthodes développées.

Bibliographie

[1] Aarts, E.H.L. and J. Korst. Simulated annealing and boltzmann machines : a stochastic approach to combinatorial and neural computing. *Wiley, Chichester*, 1989.

[2] Adiche, C. and M. Aïder. Une méthode branch and bound pour la résolution du problème d'affectation bi-objectif, Quatrième Conférence Internationale en Recherche Opérationnelle CIRO'05, 23 - 26 Mai 2005, Marrakech, Maroc.

[3] Adiche, C. and M. Aïder. New variants of the simulated annealing for solving bi-objective assignment problem, *RSTI-RIA*-Volume 22 no 2 : 237–255, 2008.

[4] Adiche, C. and M. Aïder. Hybrid method for solving the multi-objective assignment problem, International conference on Metaheuristics and Nature Inspired Computing, october 29-31 th 2008, Hammamet, Tunisie.

[5] Adiche, C. and M. Aïder. Une nouvelle variante du recuit simulé pour la résolution du problème d'affectation multi objectif, Méthodes et Outils D'Aide à la Décision, MOAD'2004, 28-29 novembre 2004, Centre universitaire Docteur Moulay Tahar, Saida. Algerie.

[6] Ben Abdelaziz, K., J. Chaouachi, and S. Krichen. A hybrid heuristic for multiobjective knapsack problems. In S. Voss, S. Martello, I. Osman, and C. Roucairol, editors, Meta-Heuristics ; Advances and trends in local search paradigms for optimization, pages 205–212. Kluwer Academic Publishers, Dordrecht, 1999.

[7] Andersen, K.A., , K. Joernstenand and M. Lind. On bicriterion minimal spanning trees : An approximation. *Computers and Operations Research, 23 :1171–1182, 1996.*

[8] Arthur, J.L. and A. Ravindran. A multiple objective nurse scheduling problem. *AIIE Transactions, 13 :55–60, 1981.*

[9] Ahuja, R.K.. Minimum cost-reliability ratio path problem. *Computers and Operations Research, 15(1) :83–89, 1988.*

[10] Arthur, J.L. and K.D. Lawrence. Multiple goal production and logistics planning in a chemical and pharmaceutical company. *Computers and Operations Research, 9(2) :127–137, 1982.*

[11] Back, T., D.B. Fogel, Z. Michalewicz, and T. Baeck. Handbook of Evolutionary Compu- tation. *Institute of Physics Publishing and Oxford University Press*, 1997.

[12] Barichard, V. and J.K. Hao. Genetic tabu search for the multi-objective knapsack problem. *Tsinghua Science and Technology, 8(1) :8–13, 2003.*

[13] Bertsekas,D.. Dynamic Programming. Prentice Hall, Englewood Cliffs, NJ, 1987.

[14] Bhaskar, K.. A multiple objective approach to capital budgeting. *Accounting and Business Research, pages 25–46, winter 1979.*

[15] Bhatia, H. L., R. Malhotra and M. C. Puri. Bi-criteria assignment problem. *Operation Research*, 19(2) :84–96,1982.

[16] Boffey, B.. Multiobjective routing problems. Top, 3(2) :167–220, 1995.

[17] Collette, Y. et P. Siarry. Optimisation multiobjectif. Eyrolles, 2002.

[18] Corne, D., M. Dorigo, and F. Glover. New ideas in optimization. *McGrawHill*, 1999.

[19] Corne,D.W. and J.D. Knowles. M-paes : a memetic algorithm for multiobjective optimization. In Proceedings of the 2000 Congres on Evolutionary Computation, pages 325–332, 2000.

[20] Current J. and Marsh M. Multiobjective transportation network design : Taxonomy and annotation. *European Journal of Operational Research, 65 :4–19, 1993.*

[21] Czyżak, P., M. Hapke, and A. Jaskiewicz Application of the Pareto simulated annealing to the multiple criteria shortest path problem. Technical Report, Poland : Politechnika Poznanska Instytut Informatyki, 1994.

[22] Czyżak, P. and A. Jaskiewicz. Pareto Simulated annealing - a metaheuristic technique for multiple objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis*, 7 :34–47, 1998.

[23] Degoutin F. et Gandibleux X. Un retour d'expérience sur la resolution de problems combinatoires bi-objectifs. 5e journée du groupe de travail Programmation Mathématique MultiObjectif (PM2O), Angers, France, 2002.

[24] Dréo J., A. Pétrowski, P. Siarry, and E. Taillard. Métaheuristiques pour l'optimisation. *Eyrolles*, 2003.

[25] Ehrgott M. and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spektrum, 22 :425–460, 2000.*

[26] Emelichev V.A. and B.A. Perepelitsa. On cardinality of the set of alternatives in discrete many-criterion problems. *Discrete Mathematics and Applications, 2(5) :461–471,1992.*

[27] Engrand P. A Multiobjective Approch Based on Simulated Annealing and its Application to Nuclear Fuel Management. In *5th International Conference on Nuclear Engineering*, pages 35–51. Nice, France, 1997.

[28] Fogel. D. Evolutionary Computation : Toward a New Philosophy of Machine Intelligence. *IEEE Press (second edition), 2000.*

[29] Fonseca C.M. and P.J. Fleming. Genetic algorithms for multi-objective optimization : formulation, discussion and generalization. In Proceedings of The Fifth International Conference on Genetic Algorithms, pages 416–423, 1993.

[30] Gandibleux X., N. Mezdaoui, and A. Freville. A multiobjective tabu search procedure to solve combinatorial optimization problems. *In Lecture Notes in Economics and Mathematical Systems, volume 455, pages 291–300. Springer, 1997.*

[31] Gandibleux, X., H. Morita and N. Katoh. Use of a genetic heritage for solving the assignment problem with two objectives. Evolutionary Multi-Criterion Optimization. *Second International Conference, EMO 2003, Faro, Portugal, April 2003 Proceedings. Lecture Notes in Computer Sciences 2632 (C. Fonseca and P. Fleming and E. Zitzler and K. Deb and L.Thiele Edts.).* pp. 43-57, Springer Verlag, Berlin.

[32] Glover F.. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research,* 13(5) :533–549, 1986.

[33] Glover F. and M. Laguna. Tabu search. *Kluwer Academic Publishers*, 1997.

[34] Geman S. and D. Geman. Stochastic relaxation, Gibbs distributions, and Bayesian restoration of image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6 :721–741, 1984.

[35] Geoffrion A.M. Proper efficiency and the theory of vector maximization. *Journal of mathematical Analysis and applications*, 1968.

[36] Goldberg D.E.. Genetic algorithms for search, optimization, and machine learning. *Reading, MA : Addison-Wesley, 1989.*

[37] Guo Y., A. Lim, B. Rodrigues, Y. Zhu. Heuristics for a bidding problem. *In Journal of Computers and Operations Research*-Volume 33 no 8 : 2179–2188, 2006.

[38] Hamacher H.W. and G.Ruhe. On spanning tree problems whith mutiple objectives. *Annals of Operations Research* 52 :209–230,1994.

[39] Hansen M.P.. Tabu search for multiobjective optimization :MOTS. *In Proceedings of 13th International Conference on MCDM, 1997.*

[40] Hao J.K., P. Galinier and M. Habib.. Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes. *Revue d'Intelligence Artifcielle*, 13(2) :283–324, 1999.

[41] Holland J.H.. Adaptation in natural and artificial systems. PhD thesis, University of Michigan Press, 1975.

[42] Ignizio J.P.. Goal Programming and Extensions.. *Lexington Books, Lexington,KY* , 1976.

[43] Jaszkiewicz, A. and R. Slowinski. Pareto Simulated Annealing for Fuzzy Multiobjective Combinatorial Optimization. *Journal of Heuristics*, 6(3) :–, 329–345, 2000.

[44] Jaszkiewicz A.. Genetic local search for multiple objective combinatorial optimization. *European Journal of Operational Research, 137(1) :50–71, 2002.*

[45] Jozefowiez N., F. Semet, et E-G. Talbi. Une méta-heuristique parallèle et hybride pour un problème de tournées de véhicules multi-critère. *Dans ROADEF'2002, 2002.*

[46] Koulamas C., S.R. Anthony, and R. Jean. A survey of simulated annealing application to operations research problems. *OMEGA*, 22 :41–56, 1994.

[47] Kouvelis and R.C. Carlson. Total unimodularity applications in bi-objective discrete optimization. *Operations Research Letters*, 11 : 61–65, 1992.

[48] Kirkpatrick, S., C. D. Gelatt and M. P. VecchiOptimization by Simulated Annealing. *Science*, 220(4598) :671–680, 1983.

[49] Korhonen P., S.Salo and R.E. Steuer. A heuristic for estimating Nadir criterion values in multiple objective linear programming. *Operations Research*, 45(5) :751–757, 1997.

[50] Knowles J., L. Thiele and E. Zitzler. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. TIK Report 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, February 2006.

[51] Lee S.M.. Goal Programming for decision analysis. *Auerbach Publications, Philadelphia,PA*, 1972.

[52] Lee, S. M and M. J. Schniederjans A multicriteria assignment problem : A goal programming approach. *Interfaces*, 13(4) :75–81, 1983.

[53] Little J.D.C., Murty K.G., Sweeney D.W., Karel C. An algorithm for the traveling salesman problem. *Operations Research*, vol. 11 : 972–989, 1963.

[54] Martins E.Q.V and J.C.N Climaco. On the determination of the nondominated paths in a multiobjective network problem. *Methods of Operations Research*, 40 :255–258, 1981.

[55] Martins E.Q.V and J.C.M. Climaco. A bicriterion shortest path algorithm. *European Journal of Operational Research*, 11 :399–404, 1982.

[56] Metropolis N., Rosenbluth A.W., Rosenbluth M.N., Teller A.H. and Teller E. Equation of State Calculations by Fast Computing Machines. *J.Chem.Phys., vol. 21 : 1087–1090, 1953.*

[57] Nagar A., J. Haddock et S. Heragu. Multiple and bicriteria scheduling : A litterature survey. *European Journal of Operational Research*, 46, –734, 1995.

[58] Nam, D. K. and C. H. Park . Multiobjective Simulated Annealing : a comparative study to evolutionary Algorithm. *Innt. J. Fuzzy Systems*, 2(2) :87–97, 2000.

[59] Osman I. and G. Laporte. Metaheuristics : A bibliography. *Annals of Operations Research*, 63 :513–623, 1996.

[60] Othmani I. Optimisation multicritère : Fondements et Concepts. PhD thesis, Université de Grenoble, 1998.

[61] Pareto V. Cours d'économie politique (1897).

[62] Park Y.B. and C.P. Koelling. An interactive computerized algorithm for multicriteria vehicle routing problems. *Computers and Industrial Engineering, 16 :477–490, 1989.*

[63] Przybylski A., Gandibleux X. and Ehrgott M. Two phase algorithm for the bi-objective assignment problem. *European Journal of Operational Research* 185(2), 509–533, 2008.

[64] Pirlot and J. Teghem. Optimisation approchée en recherche opérationnelle (Traité IC2, Série Informatique et systèmes d'information). *Hermès Sciences, 2002.*

[65] Rana K. and R.G. Vickson. A model and solution algorithm for optimal routing of a time-chartered containership. *Transportation Science, 22 :83–96, 1988.*

[66] Reeves C..Modern Heruristic Techiques for Combinatorial Problems. *Advanced Topics in Computer Science. McGrawHill, London*, 1995.

[67] Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. PhD thesis, Vanderbilt University, 1984.

[68] Schwefel H-P.. Numerical optimization of computer models. *Wiley, Chichester, 1981.*

[69] Serafini, P. Some considerations about computational complexity for multi objective combinatorial problems. In J. Jahn and W. Krabs, editors, Recent advances and historical development of vector optimization volume 294 of Lecture Notes in Economics, 1987.

[70] Serafini, P. Simulated annealing for multi objective optimization problems. In *Proceedings of the Xth International conference on MCDM*, Tapei, 1992.

[71] Serafini, P. Simulated annealing for multi objective optimization problems. In *Multiple Criteria Decision Making. Expand and enrich the domains of thinking and application*, pages 283–292, 1994.

[72] Sergienko I.V. and V.A. Perepelitsa. Finding the set of alternatives in discrete multicriterion problems. *Cybernetics, 27(3) :673–683,1991.*

[73] Siarry P., La méthode du recuit simulé : théorie et applications. *APII. 29(4-5) pages 535–561, 1995.*

[74] Smith, K. I., R. M. Everson and J. E. Fieldsend. Dominance Mesures for Multi Objective Simulated Annealing. In *Proceedings of Congress on Evolutionary Computation, CEC04*, 23–30, 2004.

[75] Srinivas N. and K. Deb. Multiobjective optimization using non dominated sorting in genetic algorithms. *Evolutionary Computation, 2(3) :221–248, 1994.*

[76] Steuer R.. Multiple Criteria Optimization : Theory, Computation and Application. *College of Business Administration University of Georgia, 1985*

[77] Suman, B. Multiobjective simulated annealing- A metaheuristics technique for Multiobjective optimization of a constrained problem. *Foundations of Computing and Decision Sciences*, 27, 171, 2002.

[78] Suman, B. Simulated annealing based Multiobjective algorithm and their application for system reliability. *Engineering Optimization*, 35, 391, 2003.

[79] Suman, B. Study of simulated annealing based Multiobjective algorithm for Multiobjective optimization of a constrained problem. *Computers and Chemical Engineering*, 28(9),1849, 2004.

[80] Suman, B. Study of self-stopping PDMOSA and performance measure in Multiobjective optimization. *Computers and Chemical Engineering*, 29,1131–1147, 2005.

[81] Suppapitnarm, A., K. A. Seffen, G. T. Parks and P. J. Clarkson. Simulated Annealing : An alternative approach to true multiobjective Optimisation. *Engineering Optimization*, 3 :59–85, 2000.

[82] Talbi E-G.. Une taxinomie des métaheuristiques hybrides. *Dans ROADEF'2000, 2000.*

[83] Taillard É. D., L.-M. Gambardella, M. Gendreau, J.-Y. Potvin. Adaptive Memory Programming : A Unified View of Meta-Heuristics. Technical report IDSIA-19-98, IDSIA, Lugano, 1998. *European Journal of Operational Research 135 (1), 1–16, 2001*

[84] Tuyttens, D., J. Teghem, P. Fortemps, and K. Van Nieuwenhuyse. Performance of the MOSA method for the bicriteria assignment problem. *Journal of Heuristics*, 6(3) :295–310, 2000.

[85] Ulungu E. L.. Optimisation combinatoire multicritére : détermination de l'ensemble des solutions efficaces et méthodes interactives. *PhD thesis, Faculté des Sciences, Université de Mons-Hainault*, 1993.

[86] Ulungu, E. L. and J. Teghem. The Two phases method : An efficient procedure to solve bi-objective Combinatorial Optimization Problems. *Foundation of Computing and Decision Sciences*, 20(2) :149–165, 1994.

[87] Ulungu E. L. and J. Teghem. Application of the two phases method to solve the bi-objective knapsack problem. *Technical report, Faculté Polytechnique de Mons, Belgium, 1994.*

[88] Ulungu, E. L., J. Teghem, and P. Fortemps. Heuristics for multi-objective combinatorial optimisation problem by simulated annealing. In *Q. Wei J. Gu Chen and S. Wang, editors, MCDM : Theory and applications*, pages 228–238. SCI-TECH Information services, 1995.

[89] Ulungu E. L. and J. Teghem. Solving multi-objective knapsack problem by a branch-and- bound procedure. In J. Climaco, editor, Multicriteria Analysis, pages 269–278. Springer Verlag, Berlin, 1997.

[90] Ulungu, E. L., J. Teghem, and C. Ost. Interactive simulated annealing in a Multiobjective framwork : application to an industrial problem. *Journal of Operational Research Society*, 49 :104, 1998.

[91] Ulungu, E. L.,P. Fortemps, and D. Tuyttens. MOSA method : a tool for solving multiobjective combinatorial decision problems. *I. J. multi- criteria decision analysis*, 8 :221–236, 1999.

[92] Vidal R.V.. Applied simulated annealing. *Lecture Notes in Economics and Mathematical Systems*, 396, 1993. Springer-Verlag.

[93] Visee M., J. Teghem, M. Pirlot, and E.L. Ulungu. Two-phases method and branch and bound procedures to solve the bi-obective knapsack problem. *Journal of Global Optimization, 12 :139–155, 1998.*

[94] Warburton A.. Aproximation of Pareto optima in multiple-objective shortest-path problems. *Operations Research, 35(1) :70 –79, 1987.*

[95] Zhou G. and M. Gen. Genetic algorithm approach on multi-criteria minimum spanning tree problem. *European Journal of Operational Research, 114 :141–152, 1999.*

[96] Zitzler E. and L. Thiele. An evolutionary algorithm for multiobjective optimization : the stength Pareto approach. *Technical report, Swiss Federal Institute of Technology (Zurich), 1998.*