Résolution approchée du problème de set packing bi-objectifs

X. Delorme[1,2], X. Gandibleux[2] et F. Degoutin[1,2]

[1] LAMIH/ROI, Université de Valenciennes, Le Mont Houy, F59313 Valenciennes Cedex 9 - FRANCE
{Xavier.Delorme, Xavier.Gandibleux, Fabien.Degoutin}@univ-valenciennes.fr
[2] INRETS-ESTAS, 20 rue Élisée Reclus, F59650 Villeneuve d'Ascq - FRANCE
{Xavier.Delorme, Fabien.Degoutin}@inrets.fr

Résumé Le problème de «set packing» (SPP) relève du domaine de l'optimisation combinatoire en variables 0-1. À notre connaissance, le SPP avec deux objectifs (biSPP) n'a pour le moment pas encore été étudié. La résolution exacte de ce problème avec un solveur générique (Cplex) n'étant possible que pour des instances de petite taille, nous nous proposons de comparer le comportement de deux heuristiques adaptées au biSPP. La première est une heuristique évolutionnaire multi-objectifs basée sur SPEA (Strength Pareto Evolutionary Algorithm). La deuxième est une adapation au cas bi-objectifs d'une métaheuristique mono-objectif basée sur GRASP (Greedy Randomized Adaptative Search Procedure). L'étude repose sur 120 instances numériques dont les fonctions objectifs présentent des caractéristiques variées. Nous rapportons et analysons les résultats obtenus avec ces deux heuristiques sur l'ensemble de ces instances. Les enseignements tirés de cette étude mettent en évidence des améliorations possibles des deux méthodes, notamment en raison de leur complémentarité.

Mots-Clefs. Optimisation combinatoire multi-objectifs ; Problème de set packing ; SPEA ; GRASP.

1 Introduction

Soit un ensemble fini d'éléments valués $I = \{1, \ldots, n\}$ et $\{T_j\}, j \in J = \{1, \ldots, m\}$ une collection de sous-ensembles de I. Un packing est un sous-ensemble $P \subseteq I$ tel que $|T_j \cap P| \leq 1, \forall j \in J$. L'objectif du problème de set packing avec deux fonctions objectifs $q \in Q = \{1, 2\}$ est de «maximiser» la valeur totale du packing obtenu. La formulation mathématique de ce problème est donnée par le modèle mathématique suivant (1) :

$$\left[\begin{array}{lll} \text{“max”}\ z^q = \sum_{i \in I} c_i^q x_i & & \forall q \in Q \\ \text{sc} \quad \sum_{i \in I} t_{i,j} x_i \leq 1 & & \forall j \in J \\ x_i \in \{0,1\} & & \forall i \in I \\ t_{i,j} \in \{0,1\} & & \forall i \in I, \forall j \in J \end{array}\right] \tag{1}$$

avec :

- un vecteur de variables $X = (x_i)$ tel que $x_i = \begin{cases} 1 \text{ si } i \in P \\ 0 \text{ sinon} \end{cases}$
- des vecteurs de données $C^q = (c_i^q)$ tel que $c_i^q =$ valeur de l'élément i dans la fonction objectif q
- une matrice de données $T = (t_{i,j})$ tel que $t_{i,j} = \begin{cases} 1 \text{ si } i \in T_j \\ 0 \text{ sinon} \end{cases}$

Le problème de set packing avec deux objectifs (biSPP) est un problème relevant du domaine de l'optimisation combinatoire multi-objectifs (MOCO). D'après Ehrgott et Gandibleux [1], aucun papier ne traite ce problème. Le biSPP est NP-difficile. Selon Garey et Johnson [2], le problème de set packing est déjà NP-difficile au sens fort dans le cas mono-objectif.

Une solution $x^\star$ du problème biSPP est *efficace* s'il n'existe aucune autre solution réalisable x telle que $z^q(x) \geq z^q(x^\star),\ \forall q \in Q$ avec au moins une inégalité stricte. Nous notons E l'ensemble des solutions efficaces. Comme c'est généralement le cas pour les problèmes MOCO, l'ensemble E se partitionne en deux sous-ensembles :

- l'ensemble SE des solutions efficaces *supportées*,
- et l'ensemble NE des solutions efficaces *non-supportées*.

La section 2 décrit les heuristiques utilisées. Les résultats obtenus sur une série d'instances sont ensuite rapportés et commentés dans la section 3. Pour finir, la section 4 présente les conclusions de cette étude et indique quelques perspectives.

2 Résolution approchée

Ainsi, afin d'obtenir une bonne approximation de l'ensemble des solutions efficaces, nous avons décidé de travailler sur la résolution approchée du biSPP. Aucune métaheuristique n'ayant été proposée jusqu'à maintenant pour le biSPP, nous avons exploré deux pistes :

- l'adaptation au biSPP d'une métaheuristique multi-objectifs efficace sur d'autres problèmes proches,
- l'adaptation au cas bi-objectifs d'une métaheurique efficace sur le SPP mono-objectif.

2.1 SPEA

Pour la première piste, nous avons utilisé la métaheuristique SPEA (Strength Pareto Evolutionary Algorithm) proposée par Zitzler [3]. SPEA est une métaheuristique évolutionaire multi-objectifs avec une évaluation basée sur le concept de dominance Pareto. Elle a été appliquée avec succès au problème de sac-à-dos multi-dimensionnel multi-objectifs. L'adaptation de SPEA au biSPP que nous utilisons a été présentée par Gandibleux et al [4]. Elle reprend le schéma général de SPEA.

La population initiale est composée de 50 individus, chaque individu correspondant à une solution réalisable. Ces solutions sont toutes différentes vis-à-vis du vecteur de décision X. Elles sont obtenues avec un algorithme glouton appliqué sur 50 directions de recherche différentes (résultant des combinaisons convexes des objectifs $z = \lambda z^1 + (1-\lambda)z^2, \forall \lambda \in \{0, \frac{1}{49}, \ldots, \frac{48}{49}, 1\}$). Lorsque cela ne permet pas d'avoir 50 solutions différentes, la population initiale est complétée avec des solutions aléatoires saturées (c'est-à-dire telles que plus aucune variable ne peut être mise à 1 sans violer au moins une contrainte). L'opérateur de croisement utilisé est un opérateur à un point sélectionné de manière aléatoire sur le gène. À chaque génération, les parents sont remplacés par leurs enfants dans la population dans 80% des cas, et ils sont conservés dans les 20% restants. L'opérateur de mutation est ensuite appliqué à toutes les solutions résultantes. Il autorise la mutation de toutes les variables avec une probabilité d'occurence de 4%. Les solutions obtenues peuvent ainsi violer certaines contraintes, mais une procédure de réparation (algorithme glouton fixant des variables à 0 tant que toutes les contraintes ne sont pas respectées) permet de les ramener dans le domaine des solutions réalisables. Une procédure de saturation (algorithme glouton fixant des variables à 1 tant que cela est possible en conservant une solution réalisable) permet d'améliorer toutes les solutions générées. Enfin, une sélection par tournois est effectuée afin de limiter la taille de la population.

De plus, afin d'améliorer les performances de l'algorithme sur le biSPP, plusieurs éléments ont été ajoutés. Ainsi, toutes les solutions potentiellement efficaces trouvées sont conservées dans la population (ce qui n'était pas le cas avec la méthode originale). De même, afin d'améliorer les performances de l'algorithme dans les régions de l'ensemble des solutions efficaces proches des solutions extrémales[3], la saturation est effectuée dans trois directions de recherche différentes pour chaque solution (z^1, z^2 et une combinaison des deux objectifs). Enfin, pour diminuer les «trous» observés dans les approximations au cours de premières expérimentations, une recherche locale basée sur des échanges 1-1 (une variable fixée à 0, une autre à 1) a été ajoutée entre chaque génération. Cette procédure est appliquée à toutes les solutions de la population.

2.2 GRASP

La deuxième procédure de résolution repose sur une méthode approchée spécifique au SPP mono-objectif (voir Delorme et al [5]) fondée sur la métaheuristique GRASP (Greedy Randomized Adaptive Search Procedure, voir Féo et Resende [6]). Nous avons encapsulé cette procédure pour l'appliquer successivement et indépendemment sur 20 directions de recherche différentes (résultant des combinaisons convexes des objectifs $z = \lambda z^1 + (1-\lambda)z^2, \forall \lambda \in \{0, \frac{1}{19}, \ldots, \frac{18}{19}, 1\}$). La méthode de résolution appliquée étant une méthode approchée, l'utilisation de cette technique peut permettre d'approximer l'ensemble des solutions efficaces supportées ou non (au contraire d'une résolution exacte qui

[3] une solution s est appelée extrémale si $s \in SE$ et $\exists q \in Q, z^q(s) = \max_{x \in SE} z^q(x)$

ne permettra d'obtenir que les solutions supportées). L'ensemble des solutions potentiellement efficaces ainsi générées sont conservées.

Basiquement, la méthode GRASP comporte deux étapes : la première étape est un algorithme *glouton aléatoire* qui génère une solution vérifiant toutes les contraintes, cette solution est améliorée dans la seconde étape par une méthode de *recherche locale*. Ces deux méthodes sont itérées jusqu'à ce qu'un *critère d'arrêt* soit satisfait.

L'algorithme glouton aléatoire utilisé part d'une solution triviale où toutes les variables sont affectées à 0. La première étape va alors successivement fixer à 1 certaines variables. Pour choisir les variables à fixer, celles-ci sont évaluées et classées. La variable est choisie aléatoirement parmi les meilleures du classement : la *liste des candidats*. La taille de la liste des candidats est contrôlée par un paramètre α. L'algorithme glouton s'arrête lorsqu'aucune variable ne peut plus être fixée à 1 sans violer une des contraintes. L'étape de recherche locale cherche de meilleures solutions dans le *voisinage* de la solution courante en appliquant les transformations suivantes : $0-1$, $1-1$, $2-1$ et $1-2$ échanges. Ces échanges sont recherchés successivement avec une stratégie de choix de la première solution améliorante trouvée.

Trois extensions complétant le squelette de la méthode GRASP sont aussi utilisées. La première appelée *«reactive GRASP»* utilise, à chaque itération des deux étapes, des statistiques sur la qualité des solutions obtenues précédemment pour choisir automatiquement la valeur appropriée pour le paramètre α. La deuxième extension vient greffer une phase d'*intensification* basée sur un *«path relinking»* [7] à chaque itération de GRASP. Le path relinking recherche un chemin entre la solution de l'itération courante et une solution précédente de bonne qualité. La construction du cheminement permet de trouver de nouvelles solutions. Enfin, la troisième extension modifie la valuation des variables afin de favoriser la sélection des variables qui évitent les contraintes les plus bloquantes. L'identification de ces contraintes se fait par un processus d'*apprentissage* statistique sur les solutions obtenues précédemment.

3 Expérimentations numériques

3.1 Instances et résolution exacte

Nous avons travaillé sur une gamme variée d'instances numériques présentant les caractéristiques numériques suivantes :

- 100 ou 200 variables,
- de 300 à 1000 contraintes,
- une densité de la matrice T allant de 1% à 3%.

De plus, six familles différentes de fonctions objectifs ont été utilisées :

A : objectifs aléatoires
$c_i^1 = rnd[1..100], c_i^2 = rnd[1..100], \forall i = 1, \ldots, n$;

B : objectifs symétriques
$c_i^1 = rnd[1..100], \forall i = 1, \ldots, n$; $c_{n-i+1}^2 = c_i^1, \forall i = 1, \ldots, n$;

C : objectifs aléatoires avec motifs
$l_1 = rnd[1..\frac{n}{10}], l_2 = rnd[1..\frac{n}{10}], \ldots;$
$c_1^1 = c_2^1 = \ldots = c_{l_1}^1 = rnd[1..100]$; $c_{l_1+1}^1 = c_{l_1+2}^1 = \ldots = c_{l_1+l_2}^1 = rnd[1..100]$; ...
(idem pour $c_i^2, \forall i = 1, \ldots, n$)
D : objectifs symétriques avec motifs
$l_1 = rnd[1..\frac{n}{10}], l_2 = rnd[1..\frac{n}{10}], \ldots;$
$c_1^1 = c_2^1 = \ldots = c_{l_1}^1 = rnd[1..100]$; $c_{l_1+1}^1 = c_{l_1+2}^1 = \ldots = c_{l_1+l_2}^1 = rnd[1..100]$; ...
$c_{n-i+1}^2 = c_i^1, \forall i = 1, \ldots, n;$
E : un objectif unitaire et un aléatoire
$c_i^1 = 1, c_i^2 = rnd[1..100], \forall i = 1, \ldots, n;$
F : un objectif unitaire et un avec motifs
$c_i^1 = 1, \forall i = 1, \ldots, n;$
$l_1 = rnd[1..\frac{n}{10}], l_2 = rnd[1..\frac{n}{10}], \ldots;$
$c_1^2 = c_2^2 = \ldots = c_{l_1}^2 = rnd[1..100]$; $c_{l_1+1}^2 = c_{l_1+2}^2 = \ldots = c_{l_1+l_2}^2 = rnd[1..100]$; ...

Cela représente un total de 120 instances numériques (20 par famille de fonctions objectifs). Toutes ces instances sont disponibles dans la collection d'instances de problèmes MOCO hébergée sur le site web de la MCDM society [8].

La frontière efficace de ces instances a pu être obtenue (voir Degoutin et Gandibleux [9]) à l'aide d'une procédure dichotomique utilisant Cplex comme solver. Ainsi, il a été observé que le nombre de solutions efficaces pour ce problème est généralement faible par rapport à d'autres problèmes combinatoires bi-objectifs classiques. Par contre, le temps CPU nécessaire pour les obtenir peut être exhorbitant (c'est-à-dire pouvant monter jusqu'à plus de 360 000 secondes sur la machine utilisée dans cette étude) malgré la taille assez petite des instances considérées.

De plus, les bornes immédiates étudiées par Ehrgott et Gandibleux [10] sur un ensemble de problèmes MOCO semblent de très mauvaise qualité pour le biSPP. Ainsi, une borne inférieure obtenue avec un algorithme glouton sur différentes directions de recherche ne donne pas une bonne description de l'ensemble des solutions efficaces et en est éloigné. La borne supérieure obtenue par la relaxation linéaire du problème fournit encore moins d'information. La figure 1 montre la frontière efficace (01) et les bornes obtenues, par relaxation linéaire (RL) et par un glouton, sur une des instances considérées.

3.2 Résultats obtenus

Nos deux algorithmes ont été testés sur un Pentium III cadencé à 800 MHz. Le critère d'arrêt utilisé était le temps de résolution qui était fixé à 20 secondes pour les instances à 100 variables et à 80 secondes pour celles à 200 variables. Les résultats indiqués correspondent aux résultats moyens observés sur 16 lancements

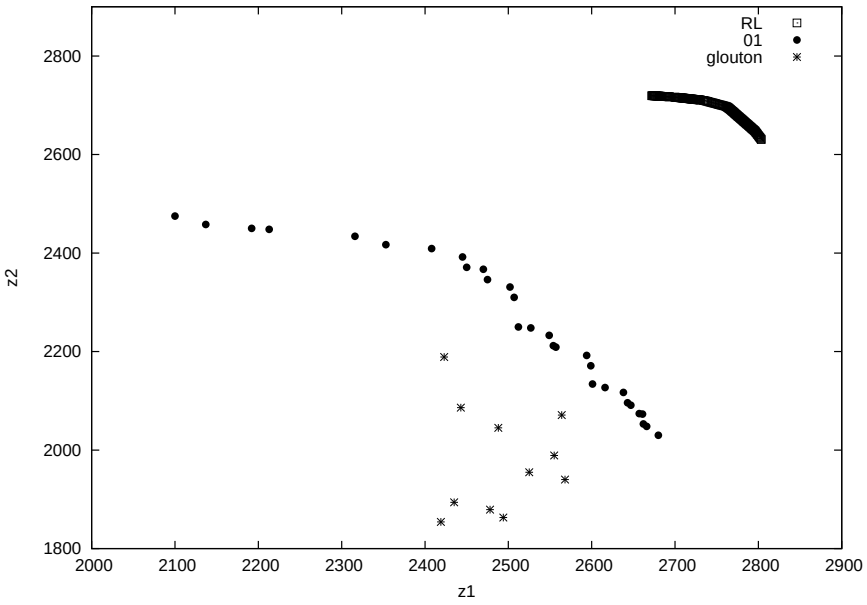

Fig. 1. Exemple de frontière efficace et de bornes

indépendants de chaque algorithme. Pour mesurer la qualité des approximations obtenues, nous avons utilisé trois indicateurs :

1. le pourcentage de solutions efficaces trouvées (M1 proposé par Ulungu et al. [11]),
2. la distance euclidienne moyenne à la frontière efficace,
3. l'hypervolume (S-metric proposé par Zitzler et Thiele [12]) correspondant pour le biSPP à la surface définie dans l'espace des critères par l'ensemble des solutions de l'approximation.

De plus, afin de faciliter la comparaison sur l'ensemble ou sur des familles d'instances, les valeurs de l'indicateur hypervolume ont été exprimées en pourcentage par rapport aux valeurs de cet indicateur pour la frontière efficace (une valeur de 100% correspond donc à une couverture complète de la frontière efficace). Tous ces résultats sont synthétisés dans les tableaux 1, 2 et 3.

Au regard de ces indicateurs, les approximations obtenues avec les deux méthodes sont de bonne qualité. Les résultats sont cependant moins bons sur les instances de plus grande taille (200 variables). À l'examen des indicateurs 2 et 3 (distance moyenne et hypervolume), les résultats de GRASP sur les instances des familles E et F semblent un peu moins bons. Ceci est probablement lié à la présence d'un objectif unitaire.

Par contre, si SPEA apparaît comme meilleur vis-à-vis des deux premiers indicateurs, GRASP produit de meilleurs résultats vis-à-vis de l'indicateur d'hy-

		A	B	C	D	E	F	Total
SPEA	100 variables	81%	88%	81%	86%	95%	87%	**86%**
	200 variables	71%	68%	71%	80%	72%	81%	**74%**
	Total	**75%**	**76%**	**75%**	**82%**	**82%**	**83%**	**79%**
GRASP	100 variables	91%	92%	85%	94%	84%	88%	**89%**
	200 variables	58%	54%	64%	67%	57%	62%	**60%**
	Total	**72%**	**70%**	**73%**	**79%**	**68%**	**73%**	**72%**

Tab. 1. Pourcentages moyens de solutions efficaces trouvées

		A	B	C	D	E	F	Total
SPEA	100 variables	4,00	1,43	1,98	0,60	0,25	0,60	**1,48**
	200 variables	5,08	6,73	6,68	3,44	3,20	1,72	**4,47**
	Total	**4,62**	**4,49**	**4,70**	**2,24**	**1,96**	**1,25**	**3,21**
GRASP	100 variables	0,44	0,71	1,21	0,54	3,12	4,77	**1,80**
	200 variables	8,52	8,53	5,42	6,05	13,61	20,30	**10,40**
	Total	**5,12**	**5,24**	**3,65**	**3,73**	**9,19**	**13,76**	**6,78**

Tab. 2. Distances moyennes à la frontière efficace

		A	B	C	D	E	F	Total
SPEA	100 variables	99,5%	99,4%	99,0%	99,6%	99,7%	99,3%	**99,4%**
	200 variables	98,5%	98,3%	98,9%	99,0%	98,6%	98,7%	**98,7%**
	Total	**98,9%**	**98,8%**	**99,0%**	**99,3%**	**99,1%**	**99,0%**	**99,0%**
GRASP	100 variables	100,0%	100,0%	100,0%	100,0%	99,7%	99,9%	**99,9%**
	200 variables	99,8%	99,7%	99,8%	99,8%	98,9%	99,0%	**99,5%**
	Total	**99,9%**	**99,8%**	**99,9%**	**99,9%**	**99,2%**	**99,4%**	**99,7%**

Tab. 3. Pourcentages moyens de l'hypervolume de la frontière efficace

pervolume. En fait, SPEA génère une population très proche de la frontière et une densité de solutions intéressante, mais éprouve des difficultés à trouver les solutions extrémales. Au contraire, l'approximation produite par GRASP représente une bonne répartition le long de la frontière efficace, mais la densité de solutions obtenues est plus faible (sans doute à cause de l'utilisation de directions de recherche). La figure 2 rapporte les approximations obtenues sur une exécution de chacune des heuristiques et illustre bien le comportement caractéristique des deux dernières.

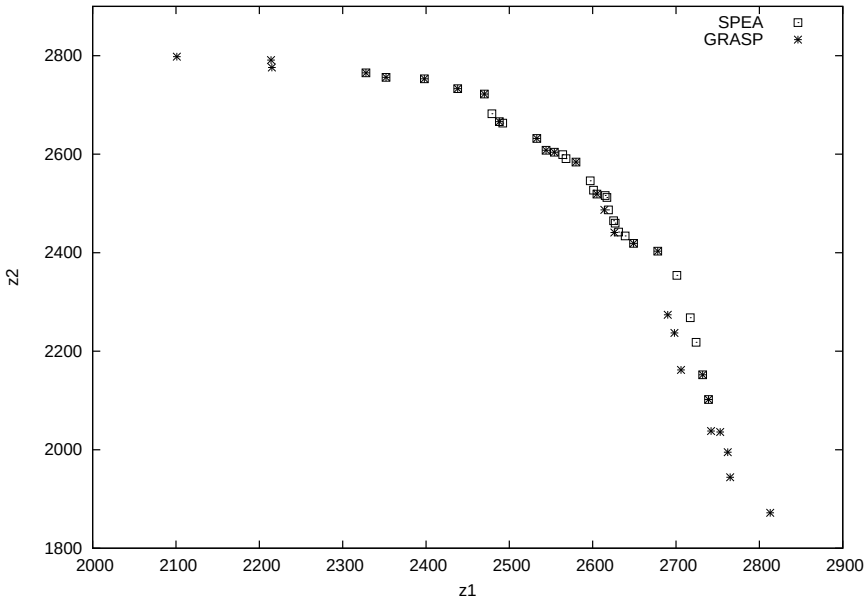

Fig. 2. Exemple d'approximations obtenues

4 Conclusion et perspectives

Dans ce papier, nous avons présenté deux méthodes heuristiques pour approximer la frontière efficace du problème de set packing bi-objectifs. Ces méthodes sont basées sur les métaheuristiques SPEA et GRASP. Les résultats obtenus semblent très intéressants pour ces deux méthodes, malgré les points faibles observés pour chacune d'elles. Cependant, il semble que des progrès peuvent encore être réalisés. Ainsi, une amélioration possible pourrait venir d'une adaptation plus fine de l'algorithme GRASP au cas bi-objectifs. Certaines phases comme le path relinking pourraient certainement être plus efficaces en étant utilisées sur des solutions générées pour des directions de recherche différentes. Une

autre possibilité serait d'utiliser GRASP pour initialiser l'algorithme SPEA. En effet, nous avons vu que les solutions obtenus avec un glouton (et qui sont utilisées comme population initiale dans SPEA) ne présentent pas une très bonne répartition le long de la frontière. L'utilisation de SPEA devrait alors s'effectuer dans un temps plus réduit pour compenser le temps supplémentaire nécessaire à l'élaboration des solutions initiales, mais cela pourrait permettre de densifier l'approximation obtenue tout en conservant une bonne répartition.

Références

1. Ehrgott, M., Gandibleux, X. : Multiobjective combinatorial optimization. In Ehrgott, M., Gandibleux, X., eds. : Multiple Criteria Optimization : State of the Art Annotated Bibliographic Survey, Kluwer's International Series in Operations Research and Management Science. Volume 52. Kluwer Academic Publishers, Boston (2002) 369–444
2. Garey, M.R., Johnson, D.S. : Computers and intractability : a guide to the theory of NP-Completeness. V.H. Freeman and Company (1979)
3. Zitzler, E. : Evolutionnary Algorithms for Multiobjective Optimization : Methods and Applications. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Suisse (1999)
4. Gandibleux, X., Degoutin, F., Delorme, X. : A first feedback on set packing problems with two objectives. Workshop on Multiple Objective Metaheuristics (MOMH), Carré des Sciences, Paris, France (2002)
5. Delorme, X., Gandibleux, X., Rodriguez, J. : GRASP for set packing problems. European Journal Of Operational Research **159** (2004) À paraître.
6. Féo, T.A., Resende, M.G. : A probabilistic heuristic for a computationally difficult set covering problem. Operations Research Letters **8** (1989) 67–71
7. Glover, F., Laguna, M. : Tabu Search. Kluwer academic publishers (1997)
8. MCDM society : Site web. (`http://www.terry.uga.edu/mcdm/`)
9. Degoutin, F., Gandibleux, X. : Un retour d'expériences sur la résolution de problèmes combinatoires bi-objectifs. 5e journée du groupe de travail Programmation Mathématique MultiObjectif (PM20), Angers, France (2002)
10. Ehrgott, M., Gandibleux, X. : Bounds and bound sets for biobjective combinatorial optimization problems. In : Multiple Criteria Decision Making in the New Millennium (M. Koksalan and St. Zionts Eds.). Lecture Notes in Economics and Mathematical Systems. Volume 507., Springer (2001) 241–253
11. Ulungu, E.L., Teghem, J., Fortemps, P. : Heuristics for multi-objective combinatorial optimisation problem by simulated annealing. In Gu, J., Chen, G., Wei, Q., Wang, S., eds. : MCDM : Theory and Applications, SCI-TECH Information Services, Windsor, UK (1995) 228–238
12. Zitzler, E., Thiele, L. : Multiobjective Evolutionary Algorithms : A Comparative Case Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation **3** (1999) 257–271