Mini-Projet MADMC

Dan MIMOUNI et Mounswif DARKAOUI

I. Introduction

On s'intéresse dans ce mini-projet à la version suivante du problème de sélection bi-objectifs (objectifs à minimiser) :
Données :

- n objets
- un entier k
- chaque objet i est valué par (c_1^i , c_2^i)
- un intervalle I = [α_{min}, α_{max}], avec α_{min} 6= α_{max}, $\alpha_{min} < 1$, $\alpha_{max} > 0$.

Solutions réalisables : tout sous-ensemble de k objets, caractérisé par un vecteur

x = ($x_1, \ldots, x_n$) (x_i = 1 si l'objet i est sélectionné, 0 sinon).
But : déterminer une solution réalisable x (dite minimax dans la suite) minimisant :

$f_I(y) = \max \{\alpha y_1 + (1 - \alpha)y_2 : \alpha \in I\}$.

où y = c(x) = ($\sum_{i=1}^{n} c_1^i x_i$, $\sum_{i=1}^{n} c_2^i x_i$) est le vecteur-coût de la solution x. Dans la suite, pour simplifier, on se contentera de déterminer l'image d'une solution minimax dans l'espace des objectifs (cette image est appelée point minimax). Le mini-projet consiste à développer deux procédures de résolution du problème, et à comparer leur efficacité.

II. Résultats préliminaires

Question 1

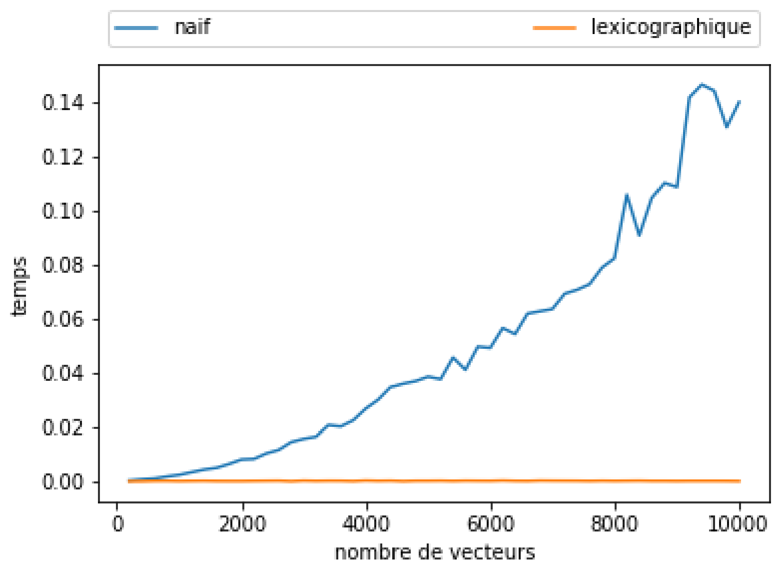
Soient y= (a,a), y' = (0,0) et y'' = (a,0) avec a < 0.
On a $f_l(y) = a$, $f_l(y') = 0$, $f_l(y+ y'') = 1+\alpha$ et $f_l(y'+y'') = \alpha$.
On a bien $f_l(y)<f_l(y')$ et $f_l(y+ y'')>f_l(y'+ y'')$.

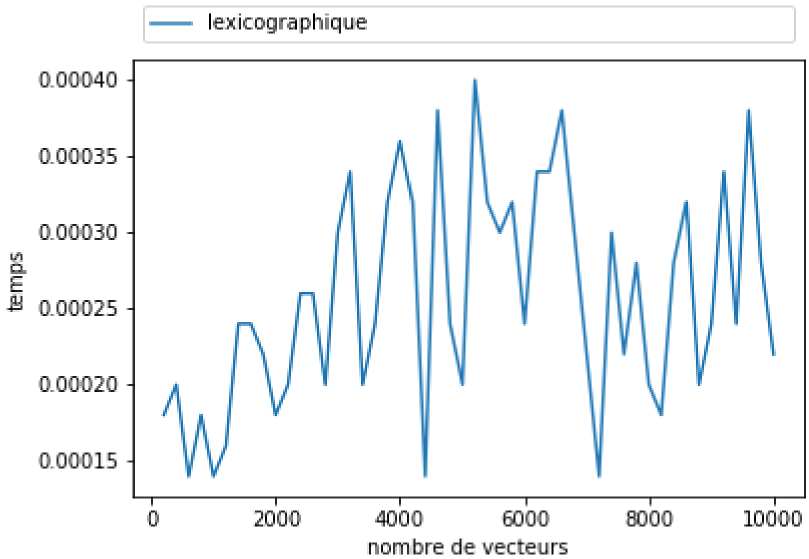
Question 5

Dans l'algo naïf, on parcours pour chaque vecteur la liste de vecteur le suivant et on compare deux à deux les vecteurs. On a donc (n-1) comparaisons pour le premier, n-2 pour le second … donc $\sum_{i}^{n} i-1$ donc O(n^2).

Dans l'algo avec tri lexicographique, on commence par trier la liste, qui se calcul en O(n*log(n)), ensuite un seul parcours suffit pour trouver les vecteurs non-dominées en O(n)
On a donc une complexité quasi-linéaire.

Pour un nombre de vecteurs variant de 200 à 10000 par pas de 200 et pour m = 1000 on obtient la courbe suivante en faisant une moyenne du temps d'exécution sur 50 ensembles tirés aléatoirement:





On peut voir que contrairement à l'algorithme lexicographique, le temps d'exécution de l'algorithme naïf augmente rapidement avec le nombre de vecteurs.

III. Une première procédure de résolution

Question 6

Supposons que l'ensemble des vecteurs non dominés ne contienne pas de solution minmax. Alors tous les vecteurs de cet ensemble ont la même image par f_I ce qui signifie qu'ils sont égaux ce qui est impossible car il s'agit d'un ensemble de vecteurs non dominés. Donc par l'absurde on peut en déduire qu'il y a toujours une solution minmax parmi les vecteurs non dominés.

Question 7

Soit E l'image des sous-ensemble Pareto optimaux de taille i dans {1,...,j}
Soit F l'image des sous-ensemble Pareto optimaux de taille i-1 dans {1,...,j-1}
Soit G l'image des sous-ensemble Pareto optimaux de taille i dans {1,...,j-1}

La relation de récurrence est la suivante E = OPT(F+f(j) U G)
Avec U : union d'ensemble, OPT : la fonction retournant l'ensemble des vecteurs non-dominés et f(j) : l'image du vecteur j par la fonction f

On initialise la procédure en mettant des vecteurs (0,0) sur la première ligne.

On a k lignes et n colonnes donc kn cellules dans le tableau
Le calcul d'une case du tableau se fait en O(m*log(m)) (cf. question 5) où M est une la nombre de vecteur par cellule(donc au plus m=n)
Donc la complexité est O(knm) (pseudo-polynomiale)

Question 8

Pour $y_1 > y_2$ on veut maximiser α donc max{$\alpha y_1 + (1 - \alpha)y_2 : \alpha \in I$} est atteint pour αmax.
Pour $y_2 > y_1$ on veut maximiser 1-α donc max{$\alpha y_1 + (1 - \alpha)y_2 : \alpha \in I$} est atteint pour αmin.

IV. Une seconde procédure de résolution

Question 10

Soit y un point non I-dominé.
Supposons qu'il existe un y' qui domine y au sens de pareto c'est à dire que $y_1'<=y_1$ ou $y_2'<=y_2$. Alors il existe un α tel que $\alpha y_1'+(1-\alpha)y_2'<=\alpha y_1+(1-\alpha)y_2$. Absurde. Donc y n'est pas dominé au sens de Pareto. Donc les points non I-dominés sont bien non dominés au sens de Pareto.

Le point minimax est défini comme min $\max\{\alpha y_1 + (1 - \alpha)y_2 : \alpha \in I\}$ on vérifie bien la définition : $\forall \alpha \in I$, $\alpha y_1 + (1 - \alpha)y_2 <= \alpha y_1' + (1 - \alpha)y_2'$. Donc si un point est minmax il est bien non I-dominé.

Question 11

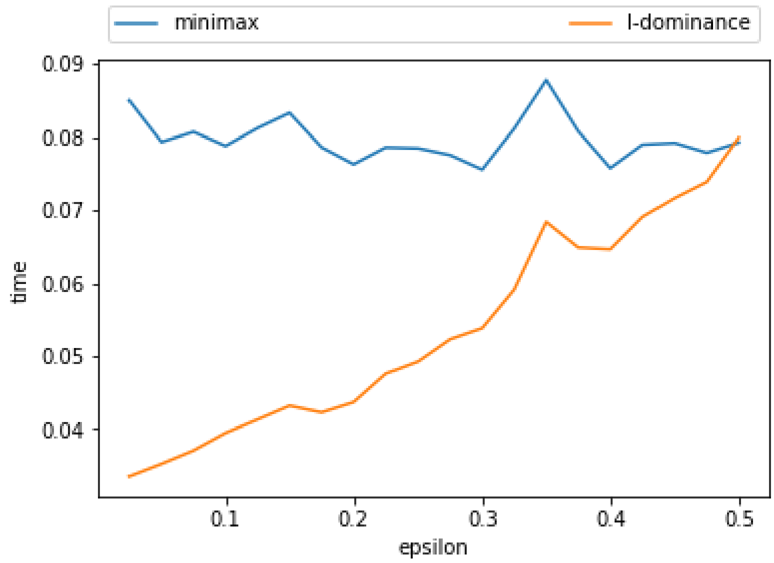
Pour tout vecteurs $y = (y_1,y_2)$ de l'instance Π on a un vecteur y' de l'instance Π' qui lui correspond, avec $y' = (\alpha_{min}y_1 + (1 - \alpha_{min})y_2, \alpha_{max}y_1 +0 (1 - \alpha_{max})y_2)$. Ainsi en trouvant les vecteurs non dominés au sens de pareto dans Π' on aura les vecteurs non I-dominés de Π.

On peut retrouver les images des vecteur dans la première instance en appliquant aux vecteurs de la deuxième instance la transformation suivante:

$y = ((1-\alpha_{max})y_1' - (1-\alpha_{min})y_2') / (\alpha_{min}-\alpha_{max}), (\alpha_{min}y_2'-\alpha_{max}y_1') / (\alpha_{min}-\alpha_{max}))$

Question 12

On compare les temps d'exécution des deux procédures (minimax et I-dominance) pour n=50, k=10, m=1000 et sur des intervalles I_ε différents de α avec $I_\varepsilon = [0.5 - \varepsilon; 0.5+\varepsilon]$ pour ε allant de 0.025 à 0.5 avec un pas de 0.025. En faisant une moyenne des temps d'exécution sur 50 instances tirées aléatoirement on obtient les courbes suivantes:



On remarque que la deuxième procédure est plus rapide pour un intervalle I petit (α_{max} -α_{min} faible) mais est aussi rapide que la première lorsque α_{max} -α_{min} est proche de 1.