Sorbonne Université le 19 Octobre 2023
MASTER ANDROIDE M1 – MOGPL

Projet MOGPL : optimisation équitable

Introduction

Les décisions prises au sein des organisations ont souvent un impact qui s'apprécie selon plusieurs points de vue. L'optimisation équitable consiste à trouver des solutions efficaces en apportant un soin particulier à préserver l'équilibre entre la satisfaction des différents points de vues considérés. Pour vous familiariser avec ce sujet, il vous est préalablement demandé de lire attentivement l'article introductif joint au dossier nommé optimequi20.pdf avant de poursuivre la lecture de l'énoncé.

De manière générale, on s'intéresse ici à un problème d'optimisation multi-dimensionnel sur un ensemble X (continu ou discret) défini en compréhension (par exemple par un système de contraintes) dont chaque solution réalisable x est évaluée par n fonctions représentant n points de vues potentiellement différents sur la qualité (ou le coût) de la solution x. Ainsi toute solution x sera évaluée par un vecteur $z(x) = (z_1(x), \ldots, z_n(x))$, $z_i(x)$ représentant la performance (ou le coût) de x selon le point de vue i. Selon les problèmes, ces points de vues pourront représenter des critères d'évaluation (décision multicritère), des évaluations individuelles de la qualité d'une solution (décision multi-agents), des évaluations dans un scénario particulier (décision dans l'incertain). Dans ce contexte, l'optimisation équitable vise à trouver une solution efficace équitable c'est-à-dire une solution que l'on ne peut améliorer sur une composante sans la dégrader sur une autre (efficacité) et qui conduit à un vecteur $z(x)$ équilibré (par exemple lorsque $n = 2$ on préfèrerait la solution (10, 10) à la solution (0, 20)).

Comme le suggère l'article mentionné ci-dessus, une façon classique d'obtenir par optimisation une solution équitable est de maximiser une moyenne pondérée ordonnée des composantes $z_i(x)$. On cherche donc une solution $x \in X$ qui maximise la fonction :

$$f(x) = \sum_{i=1}^{n} w_i z_{(i)}(x) \qquad \text{z(i) repartition equitable} \quad \text{L\_k(z)} \tag{1}$$

où w_i sont des poids positifs et décroissants lorsque i augmente ($w_i \geq w_{i+1}, i = 1, \ldots, n-1$) qui sont fixés par l'utilisateur et $(z_{(1)}, \ldots, z_{(n)})$ représente le résultat d'un tri des composantes de $(z_1(x), \ldots, z_n(x))$ par ordre croissant (ainsi $z_{(i)}(x) \leq z_{(i+1)}(x), i = 1, \ldots, n-1$.

Exemple 1 *On cherche à selectionner un ensemble de trois objets parmi 5 de manière à satisfaire équitablement deux agents sachant que les utilités des objets selon les deux agents sont respectivement* $(5, 6, 4, 8, 1)$ *et* $(3, 8, 6, 2, 5)$. *Si l'on choisit le vecteur de pondération* $w = (2, 1)$, *le problème d'optimisation équitable s'écrira :*

$$\max 2z_{(1)} + z_{(2)}$$

$$\begin{cases} z_1 = 5x_1 + 6x_2 + 4x_3 + 8x_4 + x_5 \\ z_2 = 3x_1 + 8x_2 + 6x_3 + 2x_4 + 5x_5 \\ x_1 + x_2 + x_3 + x_4 + x_5 = 3 \end{cases}$$

$$x_i \in \{0, 1\}, i = 1, \ldots, 5$$

Ce n'est pas un programme linéaire en variables 0-1 car la fonction objectif n'est pas linéaire. Il faut donc le linéariser.

1) Linéarisation de f

1.1) Pour tout vecteur $z \in \mathbb{R}$, on note $L(z)$ le vecteur $(L_1(z), \ldots, L_n(z))$ dont la composante $L_k(z)$ est définie par $L_k(z) = \sum_{i=1}^{k} z_{(i)}$. Expliquer pourquoi $L_k(z)$ est la valeur optimale du programme linéaire en variables 0-1 suivant :

$$\min \sum_{i=1}^{n} a_{ik} z_i$$

fonction objectif L_k(z)
contrainte
z_i croissant
du coup si on veut le min des sommes de z(i) , on prend les premiers i (si k = 3 , on prend 1 ,2, 3)

$$s.c. \quad \sum_{i=1}^{n} a_{ik} = k$$
$$a_{ik} \in \{0,1\}, i = 1,\ldots,n$$

1.2) On admettra que ce programme mathématique peut être relaxé en variables continues (les solutions optimales du problème relaxé sont naturellement entières) ce qui fait que $L_k(z)$ est aussi la valeur à l'optimum du programme linéaire suivant :

$$\min \sum_{i=1}^{n} a_{ik} z_i$$
$$s.c. \begin{cases} \sum_{i=1}^{n} a_{ik} = k \\ a_{ik} \leq 1 \end{cases}$$
$$a_{ik} \geq 0, i = 1,\ldots,n$$

de aik jusqu'a ank

Ecrire le dual D_k du programme linéaire ci-dessus, pour $k \in \{1,\ldots,n\}$ (on notera r_k la variables duale associée à la première contrainte du primal et b_{ik} les variables duales associées aux contraintes $a_{ik} \leq 1, i = 1,\ldots,n$. Utiliser cette formulation duale pour calculer par programmation linéaire les composantes du vecteur $L(4,7,1,3,9,2)$.

1.3) Montrer que $f(x) = \sum_{k=1}^{n} w'_k L_k(z(x))$ avec $w'_k = (w_k - w_{k+1})$ pour $k = 1,\ldots,n-1$ et $w'n = w_n$. On notera au passage que le vecteur $w' = (w'_1,\ldots,w'_n)$ à toutes ses composantes positives puisque w a ses composantes décroissantes.

1.4) En utilisant les résultats des questions précédentes la maximisation de $f(x)$ sur un ensemble X décrit par des contraintes linéaires peut s'écrire sous la forme du programme linéaire suivant :

$$\max \sum_{k=1}^{n} w'_k (k\ r_k - \sum_{i=1}^{n} b_{ik})$$
$$s.c. \begin{cases} r_k - b_{ik} \leq z_i(x),\ i = 1,\ldots,n \\ x \in X \end{cases}$$
$$b_{ik} \geq 0, i = 1,\ldots,n$$

Cette formulation linéaire utilise n^2 variables réelles positives b_{ik} pour $i,k \in \{1,\ldots,n\}$ et n variables réelles non-signées r_k, $k \in \{1,\ldots,n\}$. En utilisant cette linéarisation de f, reformuler le problème de l'exemple 1 comme un programme linéaire. Calculer la solution optimale du problème avec Gurobi.

2) Application au partage équitable de biens indivisibles

On considère un problème où n individus doivent se partager $p \geq n$ objets indivisibles de valeur connue, chaque individu ayant sa propre idée de l'utilité des objets pour lui. On suppose connue la matrice U à n lignes et p colonnes dont l'élément u_{ij} donne l'utilité de l'objet j pour l'agent i. Ces utilités sont supposées additives sur les objets c'est-à-dire que pour chaque individu i, l'utilité d'un ensemble d'objets qu'il reçoit est la somme des utilités des objets de l'ensemble. On souhaite répartir les p objets équitablement entre les n individus et pour cela on cherche l'affectation x des objets aux individus qui maximise $f(x)$ défini par l'équation (1) (les $z_i(x)$ représentant ici l'utilité de l'ensemble des objets affectés à l'individu i) pour un vecteur poids w donné à composantes strictement décroissantes.

2.1) Formuler ce problème comme un programme linéaire en variables mixtes. Utiliser cette formulation pour résoudre le problème de partage équitable de 6 objets sur 3 individus donné dans l'article (on utilisera successivement le vecteur poids $w = (3,2,1)$ puis le vecteur poids $w = (10,3,1)$ pour optimiser f) et on comparera les solutions obtenues. On comparera également ces solutions avec celle obtenue en maximisant la satsifaction moyenne des individus définie par $(z_1(x) + z_2(x) + z_3(x))/3$.

2.2) On souhaite étudier l'évolution du temps de résolution en fonction de n et p. Pour $n = 5,10,15,20$ et $p = 5n$ tirer aléatoirement 10 matrices U à coefficients entiers positifs de taille (n,p) et 10 vecteurs poids w à composantes positives strictement décroissantes et calculer le temps moyen de résolution des 10 instances pour chaque couple (n,p). Commenter les résultats obtenus.

3) Application à la selection multicritère de projets

Le responsable d'une organisation cherche à sélectionner des projets (parmi une liste de p projets soumis). Chaque projet a un coût connu $c_k, k \in \{1, \ldots, p\}$ et le coût total de l'ensemble sélectionné ne doit pas dépasser l'enveloppe budgétaire fixée à $b = \frac{1}{2}\sum_{k=1}^{p} c_k$. Les projets sont évalués quant à leur aptitude à satisfaire plusieurs objectifs de l'entreprise. On note u_{ij} l'utilité du projet j pour satisfaire l'objectif i. Ici encore ces utilités sont supposées additives et l'aptitude d'un ensemble de projets x à satisfaire l'objectif i (notée $z_i(x)$) est définie comme la somme des utilités u_{ij} des projets j selectionnés. Le but étant de trouver une solution équilibrée entre la satisfaction des différents objectifs, on cherche, ici encore, le sous ensemble x qui maximise $f(x)$ défini par l'equation (1) pour un vecteur poids w donné à composantes strictement décroissantes.

3.1) Formuler ce problème comme un programme linéaire en variables mixtes. Utiliser cette formulation pour résoudre l'instance donnée dans l'article concernant un problème bi-objectif de sélections de projets parmi 4 projets sous une contrainte de budget $b = 100$. On utilisera successivement le vecteur poids $w = (2, 1)$ puis le vecteur poids $w = (10, 1)$ pour optimiser f et on comparera les solutions obtenues. On comparera également ces solutions avec celle obtenue en maximisant la satisfaction moyenne des objectifs définie par $(z_1(x) + z_2(x))/2$.

3.2) On souhaite étudier l'évolution du temps de résolution en fonction de n et p. Pour $n = 2, 5, 10$ et $p = 5, 10, 15, 20$ tirer aléatoirement 10 matrices U à coefficients entiers positifs de taille (n, p), des coûts c_k positifs puis calculer le temps moyen de résolution des 10 instances pour chaque couple (n, p). Commenter les résultats obtenus.

4) Application à la recherche d'un chemin robuste dans un graphe

On se place dans un contexte de planification d'itinéraires dans l'incertain. Dans un réseau modélisé par un graphe orienté on recherche un chemin rapide pour aller d'un sommet initial à un sommet destination. Le problème est compliqué par le fait que plusieurs scénarios sont envisagés sur les temps de transport dans ce réseau. Plus précisément on considère un ensemble $S = \{1, \ldots, n\}$ de scénarios possibles et le temps pour parcourir chaque arc (i, j) du graphe dans les différents scénarios est donné par le vecteur $(t^1_{ij}, \ldots, t^n_{ij})$ où t^s_{ij} représente le temps nécessaire pour parcourir l'arc (i, j) dans le scénario s pour tout $s \in S$. Les temps sont additifs le long d'un chemin ce qui signifie que le temps pour parcourir un chemin P dans le scénario s est défini par $t^s(P) = \sum_{(i,j)\in P} t^s_{ij}$.

4.1) Etant donné un graphe orienté G, formuler un programme linéaire qui permette de calculer le chemin le plus rapide du sommet initial au sommet destination dans un scénario s donné, $s \in S$ (indication : on pensera à reformuler le problème comme celui de la recherche d'un flot max à coût minimum). Application : on considère un problème à deux scénarios $S = \{1, 2\}$ et le graphe de la figure 1 dans lequel les vecteurs coûts étiquetant les arcs représentent les temps de parcours dans les scénarios 1 et 2. Résoudre les programmes linéaires permettant de determiner les chemins les plus rapides de a à g dans chacun des deux scénarios.

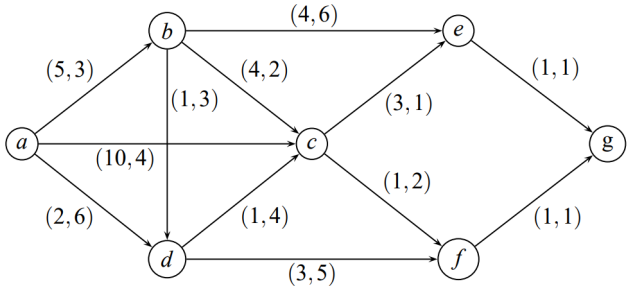

FIGURE 1 – Une instance du problème de chemin robuste à 2 scénarios

4.2) Ne sachant pas quel scénario va se produire et ne connaissant même pas leur probabilité, on va chercher un chemin robuste, c'est-à-dire qui reste rapide dans les différents scénarios. Pour cela on cherche un chemin P qui maximise $v_f(P) = f(-t^1(P), \ldots, -t^n(P))$. Proposer un programme linéaire pour calculer

un tel chemin. Appliquer le au graphe de la figure 1 pour obtenir un chemin robuste de a à g (on cherchera un chemin P qui maximise $v_f(P)$ avec le vecteur de pondération $w = (2, 1)$).

4.3) On souhaite étudier l'impact de la pondération w sur la robustesse de la solution proposée. Pour cela on utilise une famille de vecteurs de pondération $w(\alpha) \in \mathbb{R}^n$ définie pour tout $\alpha \geq 1$ par les poids :

$$w_i(\alpha) = \left(\frac{n-i+1}{n}\right)^{\alpha} - \left(\frac{n-i}{n}\right)^{\alpha}, \quad i = 1, \ldots, n$$

Pour simplifier on se placera dans le cas $n = 2$ et on tirera aléatoirement 20 fois des temps de parcours t^1_{ij} et t^2_{ij} pour les arcs (i, j) du graphe de la figure 1. Pour chacune des 20 instances ainsi générées on calculera un chemin qui maximise v_f pour le vecteur $w(1)$ (c'est-à-dire qu'on choisit $\alpha = 1$) et on représentera les 20 solutions ainsi obtenues dans le plan (t^1, t^2). On recommencera le même graphique sur les mêmes instances en optimisant f avec le vecteur poids $w(2)$ ($\alpha = 2$) puis $\alpha = 3, 4, 5$. Représenter les 5 nuages de points obtenus et analyser les résultats obtenus.

Modalités de travail et délivrables

Le travail est à effectuer en binôme constitué de deux personnes inscrites dans le même groupe de TD. Le binôme devra être déclaré par mail à votre chargé de TD (objet du mail : binome MOIARO-22) avant le lundi 7 Novembre 2022. Les projets seront déposés au plus tard le dimanche 4 décembre 2023 à minuit sur le site moodle de MOGPL. Votre livraison sera constituée d'une archive zip nommée numgroupe_nom1_nom2.zip qui comportera les sources du programme, un fichier README détaillant comment exécuter le programme, et un rapport rédigé (un fichier au format pdf nommé numgroupe_nom1_nom2.pdf) qui présentera le travail effectué et les réponses aux différentes questions. Le plan du rapport suivra le plan du sujet. Il est fortement recommandé de rédiger son rapport en LaTeX. Les projets rendus feront l'objet d'une brève soutenance lors du tme la semaine du 12 décembre 2022.

Accès au solveur Gurobi et implémentation en python

Il est fortement conseillé d'implanter vos programmes de test en python en faisant appel à *Gurobi solver* pour les résolutions de programmes linéaires (http ://www.gurobi.com/). Ce solveur est installé dans les salles de tme mais peut aussi être installé sur des machines personnelles en téléchargeant depuis une adresse de Sorbonne Université avec une licence étudiant (gratuite).