Systèmes de Décision et Préférences (SDP)
Planification de personnel et affectation de projets

Paul BÉRARD, Charlotte SASSON, Xavier JEUNOT, Ladislas LETOURNEUR

1. Introduction

La société **CompuOpti** emploie un certain nombre d'ingénieurs, avec différentes compétences, sur des projets pour des clients, sur un horizon de temps donné. Celle-ci souhaite optimiser sa planification de personnel et l'affectation des ingénieurs aux projets. Chaque projet requiert un certain nombre de qualifications et un certain nombre de jours de travail par qualification pour être réalisé. Notre but est de proposer une **organisation des emplois du temps du personnel**, sous un certain nombre de **contraintes**, de façon à optimiser **trois fonctions objectifs** : le **bénéfice financier total** (à maximiser), le **nombre maximum de projets sur lesquels un collaborateur peut être affecté** (à minimiser), et la **durée du projet le plus long** (à minimiser). Dans la suite, nous noterons (P) le **problème d'optimisation multiobjectif** sous-jacent que nous souhaitons résoudre.

Une fois des solutions respectant les contraintes trouvées, nous calculerons l'**ensemble des solutions non-dominées** de (P). Il s'agira ensuite de **choisir** parmi ces solutions non-dominées, afin de sélectionner une planification spécifique du personnel sur les différents projets. Pour cela, nous élaborerons un **modèle de préférence** prenant en entrée des exemples de plannings (partitionnés en 3 groupes : inacceptables, corrects et satisfaisants) donnés par le décideur (la personne en charge du staffing des projets, Margaux Dourtille), qui infère ce modèle de préférence et l'applique à l'ensemble des plannings non-dominés pour en extraire la **planification préférée**.

Données à disposition : `small.json`, `medium.json`, `large.json`. Veuillez consulter le code pour visualiser la forme de ces données.

Voir code sur `github.com/sassoncharlotte/optimization-project` ou cliquez ici.

2. Modélisation du problème d'optimisation multi-objectif (P) : fonctions objectifs et contraintes

Nous utilisons les notations de l'énoncé. Les notations supplémentaires ont été choisies pour être les plus explicites possibles. Nous omettons l'écriture des ensembles dans la définition des contraintes lorsqu'il n'y a pas d'ambiguïté (dans ce cas, les indices i, j, k, t sont dans $\mathcal{S}, \mathcal{J}, \mathcal{Q}, \mathcal{H}$ respectivement). Le problème d'optimisation multiobjectif s'écrit ainsi :

$$\text{maximiser} \sum_{j \in \mathcal{J}} (Y_j \times g_j - L_j \times c_j) \tag{1}$$

$$\text{minimiser} \max_{i \in \mathcal{S}} \sum_{j \in \mathcal{J}} works\_on\_job_{i,j} \tag{2}$$

$$\text{minimiser} \max_{j \in \mathcal{J}} (E_j - start\_date\_project_j) \tag{3}$$

$$\text{s.t.} \sum_{\substack{j \in \mathcal{J} \\ k \in \mathcal{Q}}} X_{i,j,k,t} \leq 1 \quad (\forall i, \forall t) \tag{4}$$

$$\sum_{\substack{j \in \mathcal{J} \\ k \in \mathcal{Q}}} X_{i,j,k,t} = 0 \quad (\forall i, \forall t \in \mathcal{V}_i) \tag{5}$$

$$X_{i,j,k,t} = 0 \quad (\forall i, \forall j, \forall k \in \mathcal{Q} \backslash (Q_i^S \cup \mathcal{Q}_j^J), \forall t) \tag{6}$$

$$Y_j \times n_{j,k} \leq \sum_{\substack{i \in \mathcal{S} \\ t \in \mathcal{H}}} X_{i,j,k,t} \quad (\forall j, \forall k \in \mathcal{Q}_j^J) \tag{7}$$

$$\sum_{\substack{i \in \mathcal{S} \\ t \in \mathcal{H}}} X_{i,j,k,t} \leq n_{j,k} \quad (\forall j, \forall k \in \mathcal{Q}_j^J) \tag{8}$$

$$X_{i,j,k,t} \times t \leq E_j \quad (\forall i, \forall j, \forall k, \forall t) \tag{9}$$

$$E_j - d_j \leq L_j \quad (\forall j) \tag{10}$$

$$\sum_{\substack{k \in \mathcal{Q} \\ t \in \mathcal{H}}} X_{i,j,k,t} \leq M \times works\_on\_job_{i,j} \quad (\forall i, \forall j) \tag{11}$$

$$\sum_{\substack{k \in \mathcal{Q} \\ t \in \mathcal{H}}} X_{i,j,k,t} - 1 \geq M \times (1 - works\_on\_job_{i,j}) \quad (\forall i, \forall j) \tag{12}$$

$$\sum_{\substack{i\in\mathcal{S}\\k\in\mathcal{Q}}} X_{i,j,k,t} \leq M' \times job\_worked\_on_{k,t} \quad (\forall j, \forall t) \quad (13)$$

$$\sum_{\substack{i\in\mathcal{S}\\k\in\mathcal{Q}}} X_{i,j,k,t} - 1 \geq M' \times (1 - job\_worked\_on_{k,t}) \quad (\forall j, \forall t) \quad (14)$$

$$start\_date\_project_j \in \mathcal{H} \quad (\forall j) \quad (15)$$

$$X_{i,j,k,t} \times t \geq start\_date\_project_j \quad (\forall i, \forall j, \forall k, \forall t) \quad (16)$$

$$X_{i,j,k,t} \in \{0,1\} \quad (\forall i, \forall j, \forall k, \forall t) \quad (17)$$

$$Y_j \in \{0,1\} \quad (\forall j) \quad (18)$$

$$E_j \in \mathcal{H} \quad (\forall j) \quad (19)$$

$$L_j \in \mathbb{N} \quad (\forall j) \quad (20)$$

On reformule (2) et (3) en linéarisant ces fonctions objectifs MinMax. On obtient donc les nouvelles fonctions objectifs (21) et (22), ainsi que les contraintes supplémentaires (23) et (24).

$$\text{minimiser } max\_nb\_of\_jobs \quad (21)$$

$$\text{minimiser } max\_duration\_of\_jobs \quad (22)$$

$$\text{s.t.} \sum_{j\in\mathcal{J}} works\_on\_job_{i,j} \leq max\_nb\_of\_jobs \quad (\forall i) \quad (23)$$

$$E_j - start\_date\_project_j \leq max\_duration\_of\_jobs \quad (\forall j) \quad (24)$$

3. Calcul de la surface des solutions non-dominées de (P)

3.1. Programmation linéaire sur Gurobi

3.1.1 Définition des constantes et des variables

- `S` : Membres du staff
- `J` : Projets
- `Q` : Qualifications
- `H` : Valeur de l'horizon du temps
- `X` : Matrice de quatre dimensions à valeurs binaires. Les dimensiosn correspondent respectivement aux :
  - Membres du staff
  - Projets
  - Qualifications
  - Jours
- `Y` : Liste de longueur égale au nombre de projets. Les valeurs sont binaires et indiquent si le projet est terminé.
- `L` : Liste de longueur égale au nombre de projets. Les valeurs sont des entiers et correspondent à la pénalité de temps pour chaque projet.
- `E` : Liste de longueur égale au nombre de projets. Les valeurs sont des entiers qui correspondent au jour de finalisation de chaque projet.
- `works_on_job` : Matrice de deux dimensions correspondant respectivement aux membres du staff et aux projets. Les valeurs `works_on_job[i, j]` sont binaires et indiquent si la personne i travaille sur le projet j.
- `max_nb_of_jobs` : Entier indiquant le nombre de projets maximal parmi les membres du staff.
- `job_worked_on` : Matrice de deux dimensions correspondant respectivement aux projets et aux jours. Les valeurs `job_worked_on[i, j]` sont binaires et indiquent si quelqu'un travaille sur le projet i le jour j.
- `max_duration_of_jobs` : Entier indiquant la durée du projet le plus long.
- `start_date_projects` : Liste de longueur égale au nombre de projets. Les valeurs sont des entiers qui correspondent au jour de démarrage de chaque projet.

3.1.2 Optimisation

Après avoir ajouté les variables au modèle Gurobi, nous avons ajouté les contraintes ainsi que les fonctions objectifs, en spécifiant que la priorité est de maximiser le bénéfice. L'optimisateur nous a fournit un ensemble de solution au sein duquel nous avons cherché les solutions non dominées grâce à la méthode ϵ-constraint.

3.2. Obtention des solutions non-dominées

La méthode ϵ-constraint permet d'obtenir les solutions non-dominées du problème d'optimisation parmi l'ensemble des solutions possibles. Pour nos besoins, la comparaison deux à deux des solutions (pour chacun des objectifs) nous a permis d'extraire l'ensemble des solutions non-dominées.

3.3. Représentation des résultats

Nous avons représenté les solutions pour les différents *datasets* sous forme de points dans un espace à 3 dimensions (les trois fonctions objectifs), en différenciant les solutions dominées (en bleu) et les non-dominées (en noir). Nous avons aussi représenté les points idéaux (en vert) et de Nadir (en rouge).

Table 1. Emploi du temps de Liam pour le projet 1 (solution 0)

	Jour 0	Jour 1	Jour 2	Jour 3	Jour 4
Compétence A	0	0	1	0	0
Compétence B	0	1	0	0	0
Compétence C	0	0	0	0	0

Table 2. Assignation des membres du personnel sur chaque projet pour la solution 0

	Job 0	Job 1	Job 2	Job 3	Job 4
Olivia	0	0	0	1	1
Liam	1	0	1	0	0
Emma	1	0	1	0	0

4. Développement d'un modèle de préférence

À partir des solutions non-dominées fournies par la résolution de (P) et de on peut mettre en place un modèle de préférences. Un modèle par pondération linéaire semble ici adapté au problème.

4.1. Modèle de préférences par pondération linéaire

Un modèle de préférences par pondération linéaire est un modèle mathématique utilisé pour représenter et quantifier les préférences d'un individu pour un ensemble de produits ou de choix. Ce modèle suppose que les préférences d'un individu pour un produit peuvent être exprimées comme une somme pondérée de ses caractéristiques. Les pondérations sont des coefficients numériques qui mesurent l'importance relative de chaque caractéristique pour l'individu.

La formule pour représenter les préférences d'un individu pour un produit i, ou reformulé pour notre problème le score de l'instance i peut être écrite comme suit :

$$S_i = \sum_{k=1}^{n} w_{i,k} x_{i,k}$$

où : S_i est le score de l'instance i, $w_{i,k}$ la pondération de l'instance i pour la caractéristique k, $x_{i,k}$ la valeur de la caractéristique k de l'instance i, n le nombre de caractéristiques considérées.

Les pondérations peuvent être déterminées à partir des données sur les préférences et les caractéristiques des produits, par exemple en utilisant des techniques d'optimisation mathématique. Une fois les pondérations déterminées, on peut utiliser la formule pour évaluer les préférences des individus pour différents produits et les utiliser pour effectuer des recommandations ou des prévisions.

Dans notre situation, un moyen pour déterminer les pondérations est de partir d'un classement arbitraire censé représenter les préférences d'un individu et d'entrainer une régression linéaire pour déterminer les poids qui correspondent à ce classement.

Nous avons donc procédé ainsi et avons déterminé un triplet de poids qui nous permet à présent d'effectuer une classification de nos solutions.

4.2. Résultat final

Ainsi, la planification retenue par notre modèle de préférence est la suivante (pour chacune des bases de données) :

Table 3. Valeurs des solutions retenues pour chaque *dataset*

	-benefice	max_nb_projets	max_duree_projet
`small`	-65.0	2.0	4.0
`medium`	-390.0	5.0	19.0
`large`	-817.0	8.0	35.0

5. Conclusion

En conclusion, notre **modélisation du problème multi-objectif** (P) avec l'utilisation de Gurobi sur Python nous a permis, dans un premier lieu, d'établir **plusieurs solutions** optimisant les fonctions objectifs et respectant les contraintes. Afin de concentrer notre choix sur un nombre plus restreint de solutions, nous avons utilisé la méthode d'ϵ-constraint pour identifier les **solutions non-dominées**, considérées comme *potentiellement* les meilleures solutions pour le décideur.

L'étape suivante a consisté à élaborer un **modèle de préférence** qui puisse aider à choisir une solution optimale parmi les solutions non-dominées. Pour ce faire, un modèle de préférence basé sur une **pondération linéaire** a été utilisé, où chaque objectif était pondéré en fonction de son importance relative pour le décideur. Une première pondération arbitraire a été choisie pour établir un classement préliminaire des solutions, puis un **modèle de régression linéaire** a été entraîné pour **extraire des poids correspondant à ce classement**. Cette nouvelle pondération a permis de créer un **score de préférence** pour chaque solution non-dominée, ce qui a permis de les classer et de sélectionner **la solution préférée du décideur**. C'est ainsi que Margaux Dourtille, grâce à notre système d'aide à la décision, a pu prendre une **décision informée pour affecter le personnel de CompuOpti à des projets de manière optimale**.