Système de Décision et Préférence

Nicolas Peruchot, Guilhem Prince, Thomas Vicaire

1 Présentation du problème

Le but de ce projet consiste à aider CompuOpti dans le choix de ses plannings de personnel et d'affectation aux projets. Chaque projet nécessite un nombre de jours donnés selon certaines compétences. Ainsi, il sera proposé une solution d'optimisation à Margaux Dourtille afin de planifier son personnel pour maximiser le résultat financier de l'entreprise, c'est à dire maximiser le bénéfice des projets.

2 Organisation générale du travail et du code

Pour travailler de manière collaborative, nous avons utilisé Github. Le code se trouve sur ce repo : `https://github.com/NicolasPeruchot/decision`.

3 Résolution du problème

Nous disposons de trois instances de taille croissante afin d'effectuer les tests, et nous utiliserons ensuite un générateur d'instance pour tester les performances de notre algorithme.

3.1 Notation

Nous utiliserons les notations suivantes pour le projet :

- $H = \{1, ..., h\}$ avec $h \in \mathbb{N} \backslash \{0\}$ horizon de temps
- $Q = \{1, ..., q\}$ est l'ensemble des qualifications
- $S = \{1, ..., n\}$ est l'ensemble du personnel
- Pour $i \in S$, un membre du personnel i est caractérisé par :
  — un sous-ensemble de qualifications $Q_i^S \subseteq Q$
  — un sous-ensemble de jours de congés $V_i^S \subseteq H$
- $J = \{1, ..., m\}$ est l'ensemble des projets
- Pour $j \in J$, un projet j est caractérisé par :
  — un sous-ensemble de qualifications $Q_j^J \subseteq Q$
  — des nombres de jours/personnes $n_{i,k} \in \mathbb{N}$ pour chaque qualification d'intérêt $k \in Q_j^J$
  — un gain $g_j \in \mathbb{N}$ obtenu le projet accompli
  — une pénalité financière par journée de retard $c_j \in \mathbb{N}$
  — une date butoire, deadline $d_j \in H$

3.2 Variables

Nous avons défini un certain nombre de variables :

- $X_{i,j,k,t} \in \{0, 1\}$ vaut 1 si la personne i réalise une qualification k pour le projet j pendant la journée t, 0 sinon, pour $i \in S$, $j \in J$, $k \in Q$, $t \in H$. Ces variables correspondent au planning du staff, et donc le livrable à rendre à CompuOpti.
- $profit \in \mathbb{N}$ correspond au profit généré par ce planning, variable que l'on cherchera à optimiser.
- max_days correspond à l'étendue maximale de jours passées à travailler sur un projet. Correspond à un objectif secondaire dans notre problème d'optimisation, pour faciliter la vie des employés de Com-

puOpti.

- max_jobs correspond au nombre maximal de projets sur lesquels une seule et même personne à travailler. Correspond également à un objectif secondaire, qui facilite aussi la vie des employés.

Variables auxiliaires pour calculer $profit$:

- $Y_j \in \{0,1\}$ vaut 1 si le projet j est réalisé totalement, 0 sinon, $j \in J$
- L_j nombre de jours de retard pour le projet $j \in J$
- E_j date de fin de réalisation du projet $j \in J$

Variables auxiliaires pour calculer max_days :

- S_j la date de début de réalisation du projet $j \in J$
- $spans_j$ étendue de ralisation du projet $j \in J$

Variables auxiliaires pour calculer max_jobs :

- $nb\_worked\_days\_per\_jobs\_and\_person_{i,j}$ le nombre de jours passés sur chaque projet par chaque personne
- $jobs\_worked\_on\_by\_person_{i,j}, (i,j) \in \{0,1\}$ vaut 1 si la personne i a travaillé sur le projet j
- $nb\_jobs\_per\_person_i$ nombre de projet auquel chaque personne a participé

3.3 Modèle

Voici les différentes équations qui régissent le modèle et leur signification.
Pour les objectifs :

— Maximiser le gain tout en minimisant les pertes financières.

$$max\ profit = max \sum_{j \in J}(Y_j \times g_j - L_j \times c_j)$$

— Minimiser l'execution en nombre de jours du projet le plus long.

$$min\ max\_days = min \max_{j \in J}\ spans_j$$

— Minimiser le nombre de projets sur lesquels un quelconque collaborateur est affecté.

$$min\ max\_jobs = min\ \max_{i \in I}\ nb\_jobs\_per\_person_i$$

Pour les contraintes :

— Une personne ne peut être affectée qu'à une seule tâche et une seule qualification par jour.

$$\sum_{j \in J, k \in Q} X_{i,j,k,t} \neq 1 \qquad \forall i \in S, \forall t \in H$$

— Une personne ne peut pas travailler sur un jour de congé.

$$\sum_{j \in J, k \in Q} X_{i,j,k,t} = 0 \qquad \forall i \in S, \forall t \in V_i$$

— Une personne ne peut pas être affecté à une qualification qu'elle n'a pas ou qui n'est pas utile au projet.

$$X_{i,j,k,t} = 0 \qquad \forall i \in S, \forall J \in J, \forall k \in Q | k \notin Q_j^S \vee k \notin Q_j^J, \forall t \in H$$

— Si un projet est réalisé, alors tous les jours de travail dédiés à chacune des qualifications intervenant dans le projet ont été couverts par des membres du personnel.

$$Y_j \times n_{j,k} \leq \sum_{i \in S, t \in H} X_{i,j,k,t} \qquad \forall j \in J, \forall k \in Q_j^J$$

— Pour un projet, il ne peut pas y avoir plus de jour travaillés que de jours nécessaires à la réalisation du projet.

$$\sum_{i \in S, t \in H} X_{i,j,k,t} \leq n_{j,k} \qquad \forall j \in J, \forall k \in Q_j^J$$

— La date de fin d'un projet est supérieure aux jours travaillés sur ce projet.

$$X_{i,j,k,t} \times t \leq E_j \qquad \forall i \in S, \forall j \in J, \forall k \in Q, \forall t \in H$$

— Le retard est supérieur à la date de fin du projet retranchée de la date de délai.

$$E_j - d_j \leq L_j \qquad \forall j \in J$$

— La date du début d'un projet est inférieure au premier jour de travail sur ce projet.

$$S_j \leq X_{i,j,k,t} * t + (1 - X_{i,j,k,t}) * h \qquad \forall i \in S, \forall j \in J, \forall k \in Q, \forall t \in H$$

— Les étendues sont égales à la différence du dernier jour de travail et du premier.

$$spans_j = E_j - S_j + 1 \qquad \forall j \in J$$

— Définition de max_days, l'étendue maximale est supérieure à toutes les étendues.

$$max\_days \geq spans_j \qquad \forall j \in J$$

— Le nombre de jours travaillés d'une personne pour un projet donné.

$$nb\_worked\_days\_per\_jobs\_and\_person_{i,j} = \sum_{k \in Q, t \in H} X_{i,j,k,t} \qquad \forall i \in I, \forall j \in J$$

— Si une personne n'a pas travaillé sur un projet, son nombre de jours passés sur ce projet est 0. Si elle a travaillé sur ce projet, son jour de nombre passés sur ce projet est au minimum 1.

$$jobs\_worked\_on\_by\_person_{i,j} = 0 \Rightarrow nb\_worked\_days\_per\_jobs\_and\_person_{i,j} = 0$$

$$jobs\_worked\_on\_by\_person_{i,j} = 1 \Rightarrow nb\_worked\_days\_per\_jobs\_and\_person_{i,j} \geq 1$$

$$\forall i \in I, \forall j \in J$$

— Le nombre de jours travaillés par une personne.

$$nb\_jobs\_per\_person_i = \sum_{j \in J} job\_worked\_on\_by\_person_{i,j} \qquad \forall i \in S, \forall j \in J$$

— Définition de max_jobs

$$\max_{i \in S}(nb\_jobs\_per\_person_i)$$

Jusque là notre modèle d'optimisation a trois fonction objectives. Or, l'implémentation en Gurobi ne peut gérer qu'un seul objectif. Notamment, si l'on dit au modèle de minimiser max_days ou max_jobs en priorité, celui ci retourne un planning totalement vide, pour assurer $max\_days = 0$ ou $max\_jobs = 0$. D'où notre envie de dire que le bénéfice reste l'objectif principal, notamment car c'est sur cela que l'entreprise CompuOpti se base et fonctionne. Pour régler ce problème, nous avons décider de créer une fonction objectif, qui prend en compte les trois objectifs, et les combine linéairement avec leur importance relative :

$$max\ 10 * profit - max\_days - max\_jobs$$

De cette manière, notre modèle cherche à maximiser en priorité le $profit$ grâce au coefficient 10, et les coefficients -1 devant max_days et max_jobs "donne la direction" de minimisation pour les objectifs secondaires (maximiser $-x$ revient à minimiser x). Par exemple, sur l'instance small, cette fonction objectif retourne 645, avec $profit = 65$, $max\_days = 2$ et $max\_jobs = 3$.

4 Résultats

4.1 Exemples de plannings générés

Avec la fonction mono-objectif cité plus haut, on obtient les plannings suivants :

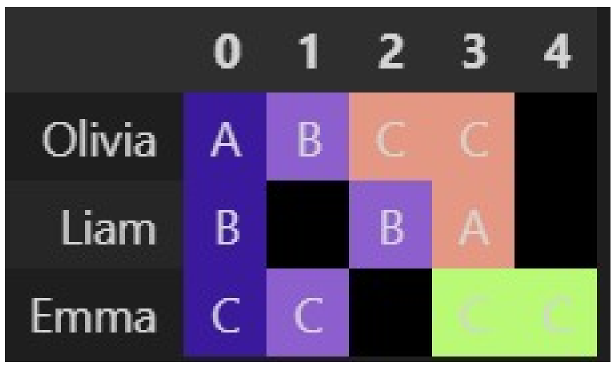

Figure 1 – Instance petite

Figure 2 – Instance moyenne

	0	1	2	3	4
Person1			Skill4	Skill2	Skill4
Person2	Skill1	Skill1		Skill2	Skill2
Person3	Skill3	Skill3	Skill3	Skill3	Skill1

Figure 3 – Instance générée 1

	0	1	2	3	4	5	6	7
Person1					Skill4	Skill4	Skill3	Skill1
Person2	Skill4		Skill4	Skill2		Skill2	Skill4	Skill4
Person3	Skill3	Skill3	Skill4		Skill3	Skill3		
Person4	Skill3	Skill3		Skill3				Skill3
Person5	Skill1	Skill1	Skill3	Skill3	Skill1	Skill3	Skill1	
Person6		Skill1	Skill3			Skill3		Skill1

Figure 4 – Instance générée 2

4.2 Surface des solutions non dominées

Pour déterminer la surface des solutions non dominées, nous avons utilisé la méthode des ϵ-contraintes. Étant donné que les deux critères $max\_days \in H$ et $max\_jobs \in J$ sont des entiers et que nous connaissons toutes leurs valeurs possibles, nous choisissons de restreindre ces deux variables à l'aide de deux contraintes en plus de celles de notre modèle :

$$max\_days \leq t \qquad t \in H$$

$$max\_jobs \leq j \qquad j \in J$$

En effectuant, une double boucle "for" imbriquée, sur les t et j, nous résolvons $(h+1) \times (m+1)$ problèmes d'optimisation mono-objectif sur le profit. Nous obtenons ainsi la surface des solutions non dominées pour le problème multi-objectif lié. Voici quelques-unes de ces surfaces :

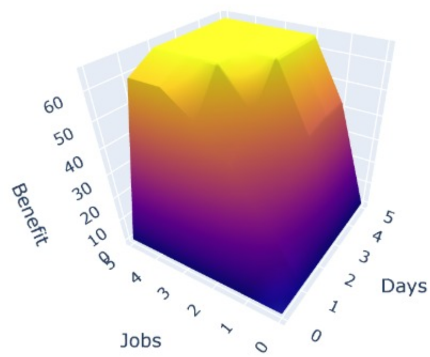

FIGURE 5 – Instance petite

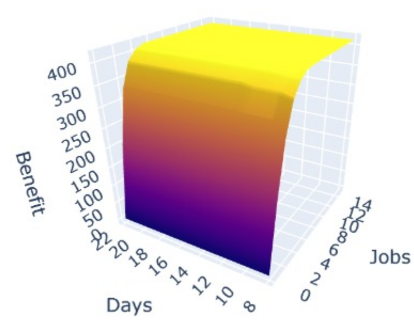

FIGURE 6 – Instance moyenne incomplète car exécution trop longue : pas de points pour $max\_jobs \leq 7$

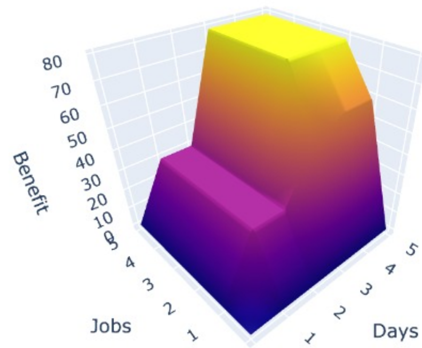

FIGURE 7 – Instance générée 1

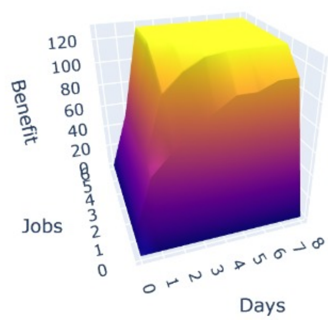

FIGURE 8 – Instance générée 2

5 Modèle de préférence

Maintenant que nous avons isolé les solutions non-dominées du problème multi-objectif, nous pouvons développer un modèle de préférence qui permettra de discriminer ces différentes solutions. Ce modèle sera basé sur les préférences que CompuOpti nous aurons fournies entre différents plannings.

5.1 Méthodologie

Supposons que l'entreprise nous transmet ses préférences sur quelques plannings, par exemple en qualifiant chaque planning comme étant soit "très

satisfaisant", "acceptable", "inacceptable". Ainsi, les solutions très satisfaisantes sont préférées à celles acceptables, elles-mêmes préférées à celles inacceptables. Pour simuler ces préférences de l'entreprise, on peut selectionner quelques solutions non-dominées (sur la surface déterminée précédemment), et leur attribuer au hasard une étiquette.
Les solutions sont chacunes uniquement déterminées par un triplet, qui est le triplet de variables $(profit, max\_days, max\_jobs)$. Nous définissons une fonction de coût f qui traduit l'ordre entre les solutions tel que :

$$f(x_1, x_2, x_3) = w_1 x_1 + w_2 x_2 + w_3 x_3$$

$$\sum_{i=1}^{3} w_i = 1, \quad w_i \in [0, 1]$$

Et ainsi,

$$solution_X \succeq solution_Y \Rightarrow w_1 x_1 + w_2 x_2 + w_3 x_3 \geq w_1 y_1 + w_2 y_2 + w_3 y_3$$

5.2 Modèle

À l'aide de cette fonction de coût, nous pouvons traduire les préférences en contraintes et donc modéliser le problème de préférences.

— Variables : w_1, w_2, w_3
— Contraintes : $\sum_{i=1}^{3} w_i = 1$ et les comparaisons entre les différentes solutions.

En résolvant ce problème, nous optenons un polyèdre de vecteurs poids admissibles : en selectionnant un vecteur poids dans ce polyèdre, les préférences sont respectées ! Nous proposons donc à CompuOpti de sélectionner un vecteur tel que w_1 soit le plus grand possible, pour donner le plus d'importance au profit.

6 Pistes d'amélioration

— Si dans ce rapport sont décrits la méthodologie et le modèle à utiliser pour gérer les préférences, le temps nous a manqué pour l'implémentation. Ainsi, si nous pouvons proposer des plannings qui semblent rentables et cohérents à CompuOpti, nous ne sommes pas en mesures de générer une liste de différents plannings idéaux (tous solutions non-dominées du problème).

— Une piste d'amélioration pour ce projet serait d'utiliser plus de force de calcul pour nous permettre de voir les résultats sur de plus grosses instances (par exemple, la grande instance fournie). De même, la surface des solutions pour l'instance moyenne n'a pas été complétée à cause du temps d'exécution. Une solution serait de faire tourner les optimisations sur des cartes graphiques GPU, et de paralléliser les optimisations dans le cas d'un calcul de surface ?