Khaldi Mohammed-Karim
Lidam Soukaina
Berrada Leila

**Projet CompuOpti**
Rapport

---

# 1 Presentation du Projet

Le but du sujet est de résoudre le problème de planification de l'entreprise CompuOpti afin de réaliser une optimisation multi-objectif, visant d'abord à maximiser les gains de l'entreprise, tout en minimsant le nombre de projets sur lequels chaque collaborateur est affecté, et en minimisant aussi le nombre de jours pour la réalisation de chaque projet.

# 2 Modélisation du problème

## 2.1 Prétraitement des données JSON

À partir du JSON représentatif des données d'entrée, on a déduit les paramètres suivants:

- **J**: le nombre de projets ( jobs ) .
- **Q**: le nombre des qualifications.
- **I**: le nombre des individus.
- **H**: l'horizon des projets.

On notera aussi :

- $H_i^{vacations}$ l'ensemble des jours de congés associés à l'individu $i$
- $Q_j^J$ l'ensemble des qualifications nécessaires pour le projet $j$
- $Q_i^I$ l'ensemble des qualifications d'un individu $i$

De plus on considère les fonctions suivantes qui seront également déduites du fichier JSON :

- $\forall j, k \ Proj(j,k) = l$ avec l le nombre d'unité de k nécessaire pour j
- $Gain(j)$ est donné par la variable gain du json
- $date\_rendu(j)$ est donné par la variable due_date du json
- $Penalite(j)$ est donné par la variable daily_penalty du json

## 2.2 Variables de décision

- Les principales variables de décision sont celle liées à **l'affectation**:

$$X_{i,j,k,t} = \begin{cases} 1 & \text{si l'individu i est affecté au job j avec la qualification k au jour t} \\ 0 & \text{sinon} \end{cases}$$

  Avec :

  - i les individus tel que $\in [0, I]$ avec I le nombre d'individus consideré.
  - j les jobs $\in [0, J]$ avec J l'ensemble des jobs.
  - k les qualifications $\in [0, Q]$ avec Q les qualifications considerées.
  - t les dates $\in [0, H]$ avec H l'horizon.

- Pour décrire la **réalisation** de projets, les variables suivantes ont été considéré initialement:

  -
$$reaProject_{j,t} = \begin{cases} 1 & \text{si le job j est fini au plus tard au jour t} \\ 0 & \text{sinon} \end{cases}$$

  -
$$rea_{t,j,k} = \begin{cases} 1 & \text{si pour le job j la qualification k est satisfaite le jour t} \\ 0 & \text{sinon} \end{cases}$$

  -
$$projet\_begin_{j,t} = \begin{cases} 1 & \text{si le job j est entamé au plus tard au jour t} \\ 0 & \text{sinon} \end{cases}$$

  Pour des raisons de temps de calcul, on a opté pour des variable plus compactes qui sont:

  -
$$Y_j = \begin{cases} 1 & \text{si le job j est réalisé} \\ 0 & \text{sinon} \end{cases}$$

  -
$$E_j = t \text{ avec t la date de fin du job j}$$

  -
$$Debut_j = t \text{ avec t la date de debut du job j}$$

- Les **retards** en jours liés aux projets sont définis par les variables entières:

$$L_j = t \text{ , avec t le nombre de jours de retard pour le projet j}$$

- Pour caractériser le **changement** de projets pour le personnel, on considère les variables suivantes:

$$shift\_change_{i,j,k,t} = \begin{cases} 1 & \text{si l'individu change de projet entre le jour t-1 et t} \\ 0 & \text{sinon} \end{cases}$$

### 2.3 Contraintes

- Contrainte d'unicité de l'affectation quotidienne du personnel :

$$\forall i, t \ \sum_{j,k} X_{i,j,k,t} \leq 1$$

- Contrainte de congé :

$$\forall i, \forall t \in H_i^{vacations}, \sum_{j,k} X_{i,j,k,t} = 0$$

- Contraintes liées aux qualifications: Un individu n'est jamais affecté à un projet pour une qualifitcation qu'il ne possède pas :

$$\forall i \forall j, \forall k \ \notin \ Q_i^I \cup Q_j^J \ \ X_{i,j,k,t} = 0$$

- Contrainte sur les affectations d'une qualification d'un projet:

$$\forall j, k \sum_{i,t} X_{i,j,k,t} \leq Proj(j,k)$$

- Contraintes pour caractériser les variables shift_change:

$$\forall i, j, k, t \ : \ X_{i,j,k,t+1} - X_{i,j,k,t} \leq shift\_change_{i,j,k,t}$$

- Contraintes pour caractériser les variables liées à la réalisation des projets:
  - Caractérisation de la date de début du projet:

$$\forall_{i,j,k,t} \ Debut_j \leq t.X_{i,j,k,t}$$

  - Caractérisation de la date de fin du projet:

$$\forall_{i,j,k,t} \ t.X_{i,j,k,t} \leq E_j$$

  - Caractérisation du retard:

$$\forall_j \ E_j - date\_rendu(j) \leq L_j$$

  - Satisfaction des qualités par projet réalisé:

$$\forall_{j,k} \ Proj(j,k).Y_j \leq \sum_{i,t} X_{i,j,k,t}$$

### 2.4 Fonctions Objectifs

- Principalement, il faut maximiser les profits :

$$\max \sum_j Gains(j).Y(j) - Penalites(j).L(j)$$

- Il faut également chercher à minimiser le nombre de changements des collaborateurs sur les differents projets:

$$\min \sum_{i,j,k,t} shift\_change(i,j,k,t)$$

- Ainsi que minimiser le nombre de jours pour réaliser les projets:

$$\min \sum_{t,j} E_j - Debut_j$$

# 3 Implementations et Solutions

## 3.1 Implementations

Au cours du projet nous avons tenté une première implémentation, néamoins au vu du nombre de variables binaires à optimiser pour gurobi, nous proposons une seconde version qui est implémenté dans notre git. Ci-dessous les idées :

### 3.1.1 Version 1

- Nous avons tout d'abord défini l'ensemble de nos variables sur Gurobi
- Ainsi que toutes les contraintes liées à notre problème
- Nous avons écrit les 3 fonctions objectifs sous Gurobi
- On optimize les 3 fonctions objectifs avec Gurobi

Cette version ne permettait pas d'obtenir les solutions des instances medium et large en un temps raisonnable. Nous proposons donc une seconde version de l'implémentation

### 3.1.2 Version 2

- On définit toutes les variables sur Gurobi
- On définit les contraintes de la fonction objectif 1 sous Gurobi
- On écrit la fonction objectif 1
- On boucle sur des valeurs maximale de shifts et de jours pour la réalisation d'un projet que l'on donne
- On ajoute les contraintes liées aux objectifs 2 et 3
- On optimise cette nouvelle formulation du problème

## 3.2 Solutions

Ainsi pour les différentes instances nous pouvons afficher la surface des solutions. De cette surface nous pouvons déduire les solutions non dominées de notre problème pour chaque instance. De plus pour chaque element dans l'ensemble de solutions non-dominées, nous pouvons afficher le planning des prévisions.

### 3.2.1 Toy Instance

Pour la petite instance, on obtient l'affichage suivant de la surface des solutions :

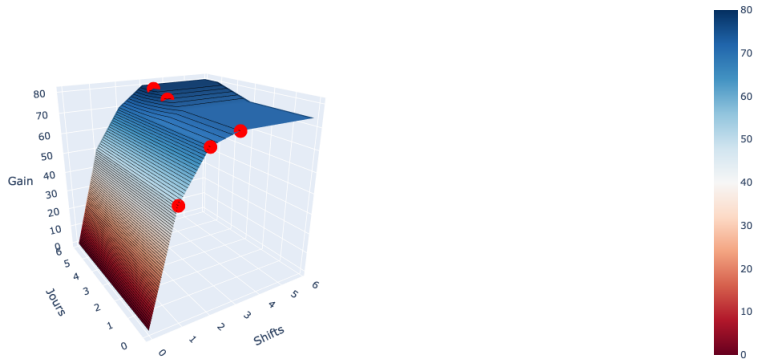


Figure 1: Affichage de la surface des solutions

On obtient donc les planning suivant pour les solutions non dominées :

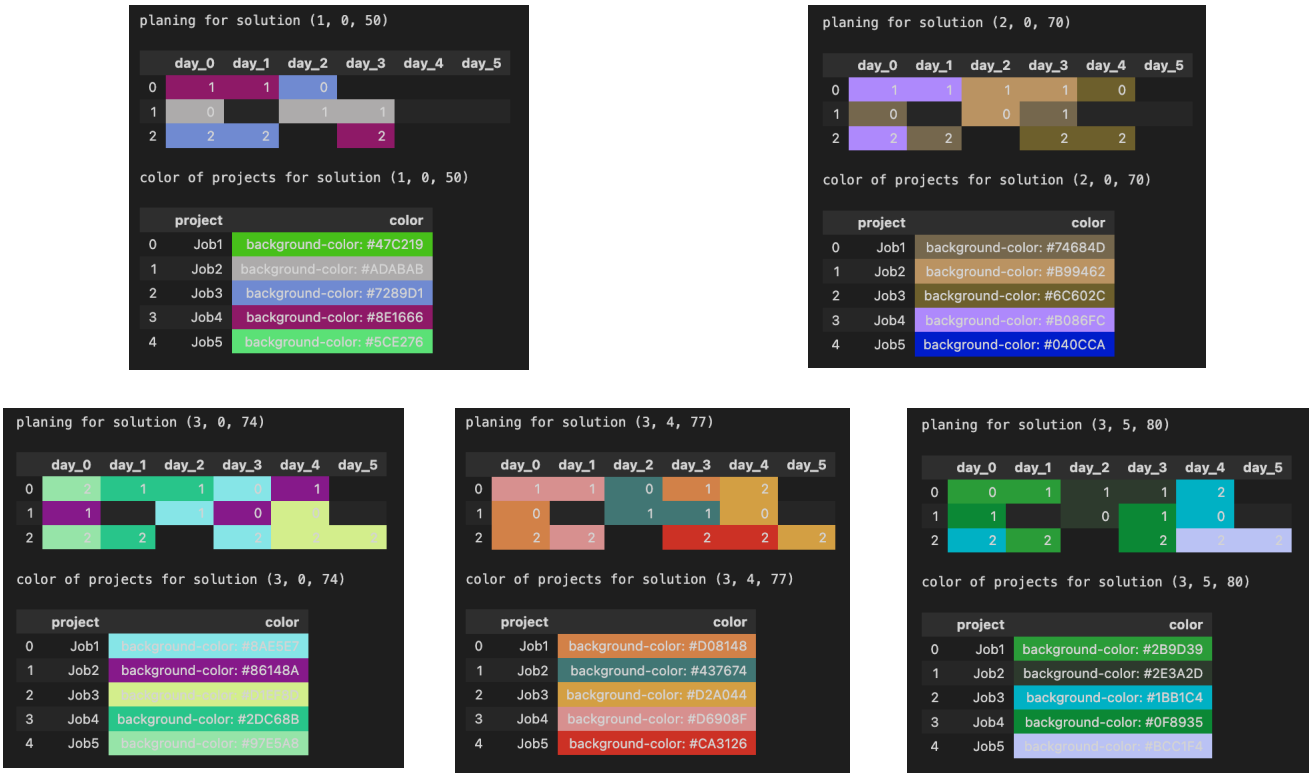


Figure 2: Plannings des solutions non dominées

### 3.2.2 Medium Instance

Pour la moyenne instance, on obtient l'affichage suivant de la surface des solutions :

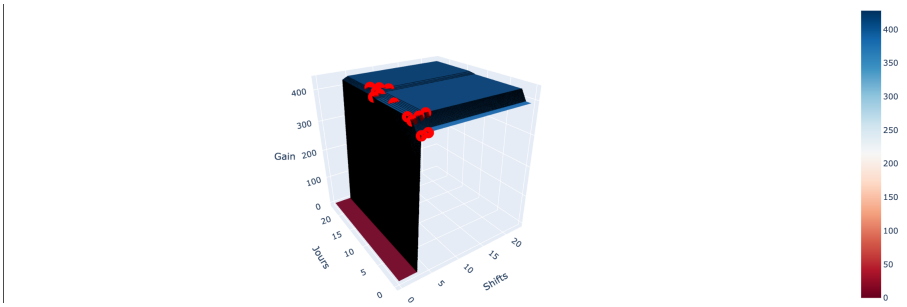


Figure 3: Affichage de la surface des solutions

Ci dessous des exemples de plannings pour les solutions non dominées :


Figure 4: Exemple de planning pour une autre solution non-dominée

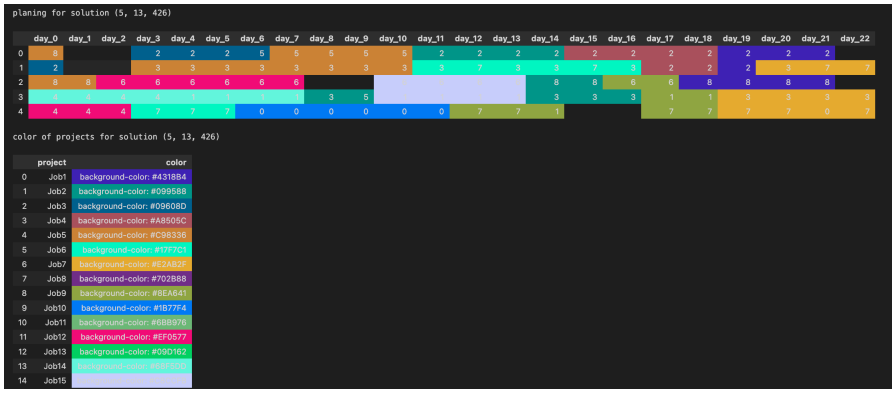


Figure 5: Exemple de planning pour l'une des solution non-dominée

### 3.2.3 Large Instance

Pour la Large instance, on obtient l'affichage suivant de la surface des solutions :

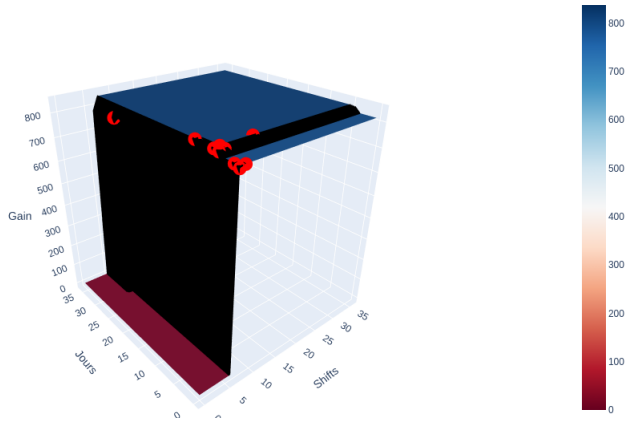


Figure 6: Affichage de la surface des solutions

# 4 Preferences

Nous nous intéressons au choix d'un système de préférences afin de classer nos solutions.
Le rendu attendu sera donc un classement des solutions obtenues. Pour ce faire, nous envisageaons de multiples méthodes de classement ( TOPSIS, Somme Pondérée ) mais au vu du fait que chacune présente ses propres limitations, nous décidons de réaliser une implémentation des méthodes citées précédemment, avec une recherche d'hyperparamètres nous semblant cohérents pour un ensemble de solutions choisies et tel que ces différentes méthodes de classement présentent des résultats cohérents entre eux et cohérents auprès de l'utilisateur final.

A cette fin, introduisons les méthodes suivantes :

- Solution 1 : TOPSIS : Technique for Order of Preference by Similarity to Ideal Solution : basé sur le calcul de distances, le principe de TOPSIS est basé sur la distance des alternatives par rapport à l'idéal positif - la meilleure valeur pour chaque critère - et l'idéal négatif. TOPSIS peut être utilisée comme une forme a posteriori pour aider le décideur à faire son choix parmi les solutions Pareto optimales.

- Solution 2 : Somme pondérée .

  Ces deux méthodes nécessitant un choix de pondérations pour nos différents objectifs, et au vu du fait que les solutions non-dominées obtenues nous semblent toutes pertinentes, nous effectuons une recherche sur la grande instance des points permettant de rendre ces deux modèles les plus similaires possibles, c'est à dire un système de poids tel que les variations entre les deux classements soient minimisées, tout en excluant évidemment les jeux de poids triviaux.

  Nous obtenons alors un ensemble de poids permettant de déduire deux ordres de préférences proches pour nos solutions non dominées :

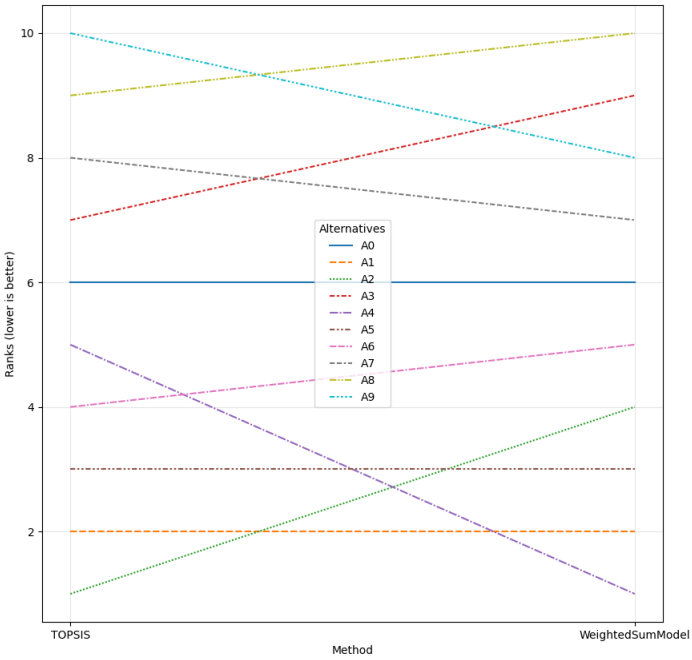


Figure 7: Comparaisons des classements via TOPSIS et via Sommes pondérées avec des poids appropriés