Recherche Opérationnelle

Modélisation et Résolution de PL/PLNE avec le solveur GLPK

par

Thomas Bocandé
Tom Audard

Groupe L1

Département Science
du Numérique

TOULOUSE-INP ENSEEIHT

Date : 4 décembre 2024

Table des matières

Table des figures

1. INTRODUCTION

Durant ce premier TP, nous avons pu découvrir le solveur GLPK. On a alors implémenté et résolu les problèmes modélisés en TD.

2. ASSEMBLAGE

2.1 Choix des variables de décision et domaine de définition

Nous avons défini les variables C pour le nombre de vélo cargo produit et S le nombre de vélo standard produit. Pour la modélisation grâce à un problème linéaire en nombre entier, C et S sont des entiers positifs. Pour la modélisation grâce à un problème linéaire, C et S sont des réels positifs, on considère alors que l'on peut ne pas produire en entier un vélo.

2.2 Éléments clefs de l'implémentions

Par rapport au travail effectué en TD, nous avons rédigé la fonction objectif et défini les trois contraintes :

- `TempsTravailMax` qui représente le temps de travail maximal de l'équipe avec le temps de production pour les 2 types de vélos.
- `EspaceStockage` qui représente l'espace de stockage disponible et les espaces utilisés pour les 2 types de vélos.
- `RessourceLimiteeCargo` qui représente la limite de vélo cargo que l'on peut assembler dû aux ressources.

Nous avons nommé `PbAssemblage.lp.txt` le fichier contenant le modèle avec un problème linéaire en nombre entier. `PbAssemblage2.lp.txt` le fichier contenant le modèle avec un problème linéaire.

2.3 Choix glpm/glpk

Pour le format de fichier du modèle, nous avons choisi le format `.lp` car le problème ne possède pas beaucoup de variables et contraintes (elles ne sont pas complexes). Il est donc assez évident de le représenter avec ce format.

2.4 Résultats et Analyses

Pour le problème linéaire en nombre entier (`SolAssemblage.sol.txt`), nous obtenons un bénéfice maximal de 438 400 €. Pour cela, le solveur nous propose de produire 232 vélos cargos et 920 vélos standards. Le résultat nous semble cohérent car un vélo cargo prends 2 et demi fois plus de place qu'un vélo standard dans l'espace de stockage et

prends plus de temps à être produit. De la limite de 700 vélos cargos produits dû aux ressources limitées est inutile dans ce problème car on ne peut pas l'atteindre. En effet

$$2.5 \times 700 = 1750 > 1500.$$

Pour le problème linéaire (`SolAssemblage2.sol.txt`), nous obtenons un bénéfice maximal de 438 461,5385 €. Très proche du problème linéaire en nombre entier, le solveur nous propose de produire 230,769 vélos cargos et 923,077 vélos standards.

```
Problem:
Rows:       5
Columns:    2
Non-zeros:  7
Status:     OPTIMAL
Objective:  Benefice = 438461.5385 (MAXimum)
```

(a) Solution au PL

```
Problem:
Rows:       3
Columns:    2 (2 integer, 0 binary)
Non-zeros:  5
Status:     INTEGER OPTIMAL
Objective:  Benefice = 438400 (MAXimum)
```

(b) Solution au PLNE

Figure 2.1 – Solutions au Problème d'assemblage

3. AFFECTATION AVEC PRISE EN COMPTE DES PRÉFÉRENCE

3.1 Choix des variables de décision et domaine de définition

Nous avons choisi une variable binaire $B_{i,j}$ qui vaut 1 si la personne numéro i participe à la tâche numéro j.

3.2 Éléments clefs de l'implémentions

Pour représenter les préférences des personnes pour chaque tâche, nous avons défini une matrice `scorestaches` qui contient le scores notés sur 10 de chaque personne pour chaque tâche.

3.3 Choix glpm/glpk

Cette fois-ci, étant donné qu'on ne connaît pas à l'avance le nombre de tâche et de personne, nous avons choisi d'utiliser le format *glpm* pour représenter le modèle. Ainsi notre modèle est plus générique et peut être résolu avec un nombre quelconque de tâche et de personne (même nombre de tâche que de personne).

3.4 Résultats et Analyses

Le solveur nous donne ainsi une satisfaction maximale de 19 (voir fichier `SolAffectation.sol.txt` ou la **Figure 3.1**) par rapport au données dans le fichier `DataAffectation.dat.txt`. Pour cela, il attribue la tâche 1 à la personne 3, la tâche 2 à la personne 1 et la tâche 3 à la personne 2. Le résultat nous semble cohérent car la personne 1 et 3 ont un score de 9 pour la tâche qui leur est attribuée et la personne 2 a alors un score de 7. Si on avait décidé d'attribuer la tâche 2 à la personne 2 (score de 8), on aurait plus perdu en satisfaction sur la personne 1 et 3 qui ont des scores bas dans les 2 autres tâches, comme vu sur la **Figure 3.2**)

```
Problem:
Rows:       6
Columns:    9 (9 integer, 9 binary)
Non-zeros:  18
Status:     INTEGER OPTIMAL
Objective:  BonheurTotal = 19 (MAXimum)
```

Figure 3.1 – Solution au Problème d'affectation

	T_1	T_2	T_3
P_1		x	
P_2			x
P_3	x		

Figure 3.2 – Tableau d'affectation des tâches

4. APPLICATION À L'E-COMMERCE

4.1 Choix glpm/glpk

Pour cette partie nous avons décider d'utiliser le format *glpm* car le problème nous propose différent cas particulier qui ont des point en commun et nous permet donc de réutiliser des bout de code ou de modélisation.

4.2 Cas Particulier 1.1

4.2.1 Choix des variables de décision et domaine de définition

Afin de mieux prendre en compte les demandes, les stocks ainsi que les coûts des fluides, nous avons décidé d'exprimer notre variable sous la forme $F_{i,j,k} \in \mathbb{R}^+$. $F_{i,j,k}$ est donc le fluide i pris dans le magasins j pour la demande k.

4.2.2 Éléments clefs de l'implémentions

Nous avons défini les matrices `demandesfluides`, `stocksfluides` et `coutsfluides` qui correspondent aux tableaux (a), (b) et (c), notée $\mathcal{D}$, $\mathcal{S}$ et C.

La fonction objectif s'écrit alors de la manière suivante :

$$\min \sum_{i,j,k} F_{i,j,k} \times C_{i,j}$$

Afin de respecter au mieux le problème nous avons défini les contraintes suivante :

$$\forall i, j, \sum_{k} F_{i,j,k} \leq \mathcal{S}_{i,j} \tag{4.1}$$

$$\forall i, k, \sum_{j} F_{i,j,k} = \mathcal{D}_{i,k} \tag{4.2}$$

La contrainte (4.1) provient des limitations de stock des différents magasins.

La contrainte (4.2) quand à elle permet le respect des demandes des clients.

4.2.3 Résultats et Analyses

Comme le montre la **Figure 4.1** le coût minimal obtenu est de 9,5 €.

```
Problem:
Rows:       10
Columns:    12
Non-zeros:  24
Status:     OPTIMAL
Objective:  CoutsTotaux = 9.5 (MINimum)
```

Figure 4.1 – Solution au cas 1.1

4.3 Cas Particulier 1.2

4.3.1 Choix des variables de décision et domaine de définition

Pour les colis, On reprend la variable $F_{i,j,k} \in \mathbb{R}^+$ et on ajoute la variable $B_{i,k}$ binaire, qui est vraie si si le magasin j fourni la demande k.

4.3.2 Éléments clefs de l'implémentions

Nous avons gardé les matrices (a) et (b) (`demandesfluides`, `stocksfluides` toujours notées $\mathcal{D}$ et $\mathcal{S}$), mais nous remplaçons la matrice (c) `coutsfluides` par les matrices (d) et (e), `coutsfixe` et `coutsvariable` (notées $\mathcal{F}$ et $\mathcal{V}$), pour modéliser les modifications du modèle.

Cela nous donne une nouvelle fonction objectif :

$$\min \sum_{i,j,k} F_{i,j,k} \times \mathcal{V}_{j,k} + B_{j,k} \times \mathcal{F}_{j,k}$$

Nous avons donc les contraintes suivante :

Les contraintes (4.1) et (4.2) du cas particulier 1.1

et

$$\forall i, j, k, M \times B_{j,k} \geq F_{i,j,k} \tag{4.3}$$

où M est un paramètre arbitrairement grand afin d'exprimer la contrainte suivante

$$B_{j,k} = \begin{cases} 0 \text{ si } \forall i, F_{i,j,k} = 0 \\ 1 \text{ sinon} \end{cases}$$

On le prendra ici égal à 1000. Cette contrainte nous permet de pouvoir compter les coûts fixe dans la fonction objectif.

4.3.3 Résultats et Analyses

Nous obtenons cette fois ci un coût minimal pour les livraison de 280 €, comme le montre la **Figure 4.2**.

```
Problem:
Rows:       22
Columns:    18 (18 integer, 6 binary)
Non-zeros:  48
Status:     INTEGER OPTIMAL
Objective:  CoutsTotaux = 280 (MINimum)
```

FIGURE 4.2 – Solution au cas 1.2

4.4 Cas Particulier 2

Nous avons ici à faire au cas du *Voyageur du Commerce*. On peut représenter le problème par un graphe pondéré où les sommets sont les lieux atteignable par le livreur et les poids sont les distances entre chaque lieux.

4.4.1 Choix des variables de décision et domaine de définition

Pour ce dernier cas particulier nous avons eu besoin de re exprimer les variables. Nous avons alors pris la variable $C_{i,j,k}$ binaire, représentant le chemin de i à j à l'étape k si vrai. Comme nous le verront par la suite, il n'est pas nécessaire de contraindre le départ du livreur de l'étape $k = 1$ au magasins *ALPHA*. En effet le livreur réalise un circuit (il passe une fois par tout les clients et revient à son point de départ).

4.4.2 Éléments clefs de l'implémentions

Ici les seules données que nous possédons sont les distances entre les clients et le magasin *ALPHA* que nous mettrons sous la forme d'une matrice `distances` notée d.

Nous avons donc la fonction objectif suivante :

$$\min \sum_{i,j,k} C_{i,j,k} \times d_{i,j}$$

Ainsi pour le problème posé nous avons les contraintes suivantes :

$$\forall i, \sum_{j,k} C_{i,j,k} = 1 \tag{4.4}$$

$$\forall j, \sum_{i,k} C_{i,j,k} = 1 \tag{4.5}$$

$$\forall k, \sum_{i,j} C_{i,j,k} = 1 \tag{4.6}$$

et

$$\forall k, \sum_{i} C_{i,n,k} = \sum_{j} C_{n,j,k+1} \tag{4.7}$$

Le contrainte (4.4) traduit le fait que chaque sommet soit lieux d'arrivée 1 fois, la (4.5) traduit le fait qu'ils doivent être lieux de départ 1 fois et la (4.6) qu'il y est une seule

transition par étape.

La contrainte (4.7) quant à elle assure que l'arrivée à l'étape k soit le départ à l'étape $k+1$.

4.4.3 Résultats et Analyses

Ici la distance minimal est 22 et la solution proposé correspond bien à un chemin réalisable qui est le suivant :

$$\alpha \rightarrow C_2 \rightarrow C_3 \rightarrow C_5 \rightarrow C_4 \rightarrow C_1 \rightarrow \alpha$$

Résultat cohérent car compris entre le chemin le court ne respectant pas les contrainte de distance égale à 6 et le chemin de distance la plus longue en prenant les plus grandes distances de distance égale à 72.

```
Problem:
Rows:       48
Columns:    216 (216 integer, 216 binary)
Non-zeros:  1008
Status:     INTEGER OPTIMAL
Objective:  Distance = 22 (MINimum)
```

Figure 4.3 – Solution au cas 2

5. CONCLUSION

Ce TP nous a permis de mieux découvrir et comprendre la modélisation de problème en recherche opérationnelle. Nous avons vu deux méthodes différentes de modéliser, l'une ou l'autre étant plus facilement applicable selon les cas. Le format .lp pour les cas précis avec des variables définies et le format glpm pour les cas générique.