Rapport recherche opérationnelle TP Sujet 1

CANON Ayoub, CORBELLARI Nolan

Département Sciences du Numérique - Deuxième année
2023-2024

Contents

Introduction et spécification des choix

Nous allons dans ce rapport traiter les différents problèmes vus en TD. GLPK nous fournis une solution à partir des contraintes que nous lui fournissons, cependant nous ne pouvons pas savoir à la main si la solution est optimale ou non. Nous avons donc fait le choix de nous appuyer sur le déroulement de la solution ou des propriétés tel que les bornes inférieures et supérieures afin de justifier qu'une solution est cohérente.

Assemblage

Dans cette section nous allons traiter le problème de l'assemblage.

Tout d'abord nous avons choisi le format `.lp` car le cas ici est très spécialisé et ne comporte que peu de données.

Le modèle est PLNE pour une semaine isolée avec des contraintes précises. En PL, on considère la semaine dans le contexte d'une succession, permettant des valeurs fractionnaires.

Variables

Les variables du problèmes sont les suivantes :

- p_{vc} : production de vélo cargo par semaine
- p_{vs} : production de vélo standard par semaine

Fonction Objectif

La fonction à maximiser dans le problème est la fonction :

$$max\ f(p_{vc}, p_{vs}) \ = \ max(700p_{vc} \ + \ 300p_{vs})$$

Contraintes

Les contraintes de ce problème sont les suivantes:

- $0 \ \leq \ p_{vc} \ \leq \ 700$
- $2,5p_{vc} \ + \ p_{vs} \ \leq \ 700$
- $\dfrac{(p_{vc} + p_{vs})11}{3} \ \leq \ 60$

Domaine

Les domaines des variables sont :

Dans le cas du modèle PLNE :

- $p_{vc} \in \mathbb{N}$
- $p_{vs} \in \mathbb{N}$

Dans le cas du modèle PL :

- $p_{vc} \in \mathbb{R}_+$
- $p_{vs} \in \mathbb{R}_+$

Résultats

Voici le résultat obtenu par GLPK pour ce problème

Pour le modèle PLNE :

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

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 HeureParSemaine
                                59.92                          60
     2 Parking                   1500                        1500
     3 ProdMaxCargo               232                         700

   No. Column name       Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 vc           *            232             0
     2 vs           *            920             0

Integer feasibility conditions:

KKT.PE: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

KKT.PB: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

End of output
```

Pour le modèle PL :

```
 Problem:
 Rows:       3
 Columns:    2
 Non-zeros:  5
 Status:     OPTIMAL
 Objective:  f = 438461.5385 (MAXimum)

    No.   Row name   St   Activity     Lower bound   Upper bound
Marginal
 ------ ------------ -- ------------- ------------- -------------
-------------
      1 HeureParSemaine
                     NU            60                           60
769.231
      2 Parking      NU          1500                         1500
261.538
      3 ProdMaxCargo B        230.769                          700

    No. Column name  St   Activity     Lower bound   Upper bound
Marginal
 ------ ------------ -- ------------- ------------- -------------
-------------
      1 vc           B        230.769             0
      2 vs           B        923.077             0

 Karush-Kuhn-Tucker optimality conditions:

 KKT.PE: max.abs.err = 7.11e-15 on row 1
         max.rel.err = 5.87e-17 on row 1
         High quality

 KKT.PB: max.abs.err = 0.00e+00 on row 0
         max.rel.err = 0.00e+00 on row 0
         High quality

 KKT.DE: max.abs.err = 0.00e+00 on column 0
         max.rel.err = 0.00e+00 on column 0
         High quality

 KKT.DB: max.abs.err = 0.00e+00 on row 0
         max.rel.err = 0.00e+00 on row 0
         High quality

 End of output
```

On voit que le maximum est atteint pour une valeur de $f = 438400$ avec comme conditions :

- HeureParSemaine = 59.92
- Parking = 1500

- ProdMaxCargo = 232
- 232 vélos de type cargo vendus
- 920 vélos de type standard vendus

Respectivement, pour le modèle PL :

- HeureParSemaine = 60
- Parking = 1500
- ProdMaxCargo = 230.769
- 230.769 vélos de type cargo vendus
- 923.077 vélos de type standard vendus

On peut à présent se demander si ces solutions sont réalistes.
On remarque que les bornes sur les contraintes sont respectées.

Affectation avec prise en compte des préférences

Dans cette section nous allons traiter le problème de l'affectation avec prise en compte des préférences.
Tout d'abord nous avons choisi le format `.mod` et `.dat` car on souhaite pouvoir généraliser ce cas à N personnes.
De plus ici les données sont importantes et variables (matrice de de taille N^2).
Ainsi il est plus judicieux d'utiliser un format `dat` pour pouvoir facilement modifier/importer les données.

Variables

Les variables du problèmes sont les suivantes :

- A_{tp} : Matrice tâche-préférences

Fonction Objectif

La fonction à maximiser dans le problème est la fonction :

$$max \sum_{i=1}^{n} \sum_{j=1}^{n} A_{tp}(i,j) * c(i,j)$$

Contraintes

Les contraintes de ce problème sont les suivantes:

- $\forall j \in [1,..,n],\ \sum_{i=0}^{n} A_{tp}(i,j) = 1$
- $\forall i \in [1,..,n],\ \sum_{j=0}^{n} A_{tp}(i,j) = 1$

Domaine

Les domaines des variables sont :

- $A_{tp} \in\ \mathcal{M}_n(\mathbb{B})$

Résultats

Voici les résultats obtenus par GLPK pour ce problème.

Exemple 1

Voici la matrice de contraintes de l'exemple 1:

$$\begin{bmatrix} 4 & 7 & 9 \\ 9 & 8 & 3 \\ 2 & 1 & 2 \end{bmatrix}$$

et la maximisation est atteinte pour la matrice binaire suivante

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

On voit que la contrainte sur les lignes et la contrainte sur les colonnes est respectées. Cette solution est donc une solution cohérente

Exemple 2

Voici la matrice de contraintes de l'exemple 2:

$$\begin{bmatrix} 4 & 7 & 8 & 2 & 10 \\ 1 & 5 & 10 & 9 & 7 \\ 10 & 7 & 7 & 5 & 8 \\ 10 & 2 & 2 & 7 & 4 \\ 2 & 9 & 6 & 4 & 9 \end{bmatrix}$$

et la maximisation est atteinte pour la matrice binaire suivante

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

On voit que la contrainte sur les lignes et la contrainte sur les colonnes est respectées. Cette solution est donc une solution cohérente.

Exemple 2

Voici la matrice de contraintes de l'exemple 2:

$$\begin{bmatrix} 10 & 7 & 4 & 1 & 10 & 10 & 9 & 1 & 4 & 7 \\ 8 & 9 & 1 & 6 & 2 & 4 & 2 & 4 & 1 & 8 \\ 3 & 5 & 3 & 1 & 1 & 3 & 9 & 8 & 4 & 6 \\ 4 & 3 & 8 & 7 & 2 & 3 & 4 & 7 & 10 & 9 \\ 6 & 10 & 6 & 3 & 6 & 7 & 3 & 1 & 5 & 6 \\ 7 & 10 & 2 & 10 & 8 & 10 & 4 & 6 & 10 & 6 \\ 7 & 6 & 5 & 2 & 7 & 2 & 10 & 5 & 5 & 2 \\ 9 & 10 & 4 & 6 & 4 & 4 & 2 & 9 & 9 & 8 \\ 1 & 6 & 7 & 4 & 1 & 8 & 6 & 9 & 9 & 1 \\ 1 & 10 & 10 & 4 & 2 & 1 & 8 & 3 & 6 & 10 \end{bmatrix}$$

et la maximisation est atteinte pour la matrice binaire suivante

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

On voit que la contrainte sur les lignes et la contrainte sur les colonnes est respectées. Cette solution est donc une solution cohérente.

Cas particulier 1.1

Variables

Les variables du problème sont les suivantes:

- q_{mfd} : Quantité de produit f stockée dans le magasin m pour la demande d.

Fonction Objectif

Le problème vise à minimiser le coût total, défini par la fonction objectif suivante :

$$min \sum_{m=1}^{nbM} \sum_{f=1}^{nbF} \sum_{d=1}^{nbD} q_{mfd} \cdot COUTS_{mf}$$

Contraintes

Les contraintes du problème sont les suivantes :

- Demande Respectée : $\forall d \in [1, \ldots, nbD], \forall f \in [1, \ldots, nbF], \text{DEMANDES}_{df} \leq \sum_{m=1}^{nbM} q_{mfd}$
- Stock Respecté : $\forall m \in [1, \ldots, nbM], \forall f \in [1, \ldots, nbF], \sum_{d=1}^{nbD} q_{mfd} \leq \text{STOCK}_{mf}$

Domaine

Les domaines des variables sont les suivants :

- $$\forall m \in [1, \ldots, nbM], \forall f \in [1, \ldots, nbF], \forall d \in [1, \ldots, nbD],\ q_{mfd} \geq 0$$

Conclusion

Voici les données du problème posé :

- le nombre de fluide est égal à 2
- le nombre de magasin est égal à 3
- le nombre de demandes est égal à 2

- Le tableau de stock de fluide

	Fluide 1	**Fluide 2**
Magasin 1	2.5	1
Magasin 2	1	2
Magasin 3	2	1

- le tableau des demandes de fluides

	Fluide 1	**Fluide 2**
Demande 1	2	0
Demande 2	1	3

- le tableau des coûts unitaires par magasin

	Fluide 1	**Fluide 2**
Magasin 1	1	1
Magasin 2	2	3
Magasin 3	3	2

Avec ces données, nous obtenons le résutat suivant :

```
Problem:    ECommerceCP11
Rows:       11
Columns:    12
Non-zeros:  36
Status:     OPTIMAL
Objective:  coutTotal = 9.5 (MINimum)

   No.   Row name   St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- -------------
   -------------
     1 coutTotal    B            9.5
     2 demandeRespectee[1,1]
                    NU            -2                          -2
   -2
     3 demandeRespectee[1,2]
                    B              0                          -0
     4 demandeRespectee[2,1]
                    NU            -1                          -1
   -2
     5 demandeRespectee[2,2]
                    NU            -3                          -3
   -3
     6 stockRespecte[1,1]
                    NU           2.5                         2.5
   -1
     7 stockRespecte[1,2]
```

```
                    NU            1                            1
   -2
     8 stockRespecte[2,1]
                    B           0.5                            1
     9 stockRespecte[2,2]
                    B             1                            2
    10 stockRespecte[3,1]
                    B             0                            2
    11 stockRespecte[3,2]
                    NU            1                            1
   -1

   No. Column name  St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- -------------
   -------------
     1 q[1,1,1]     B             2             0
     2 q[1,1,2]     B           0.5             0
     3 q[1,2,1]     NL            0             0
    3
     4 q[1,2,2]     B             1             0
     5 q[2,1,1]     NL            0             0                         <
   eps
     6 q[2,1,2]     B           0.5             0
     7 q[2,2,1]     NL            0             0
    3
     8 q[2,2,2]     B             1             0
     9 q[3,1,1]     NL            0             0
    1
    10 q[3,1,2]     NL            0             0
    1
    11 q[3,2,1]     NL            0             0
    3
    12 q[3,2,2]     B             1             0

Karush-Kuhn-Tucker optimality conditions:

KKT.PE: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

KKT.PB: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

KKT.DE: max.abs.err = 0.00e+00 on column 0
        max.rel.err = 0.00e+00 on column 0
        High quality

KKT.DB: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality
```

```

End of output
```

Donc GPLK propose le scénario suivant :

1. Pour la demande 1, on va chercher dans le magasin 1 2 unités de $fluide_1$
2. Pour la demande 2, on va chercher dans le magasin 1 0.5 unité de $fluide_1$
3. Pour la demande 2, on va chercher dans le magasin 1 1 unité de $fluide_2$
4. Pour la demande 2, on va chercher dans le magasin 2 0.5 unité de $fluide_1$
5. Pour la demande 2, on va chercher dans le magasin 2 2 unités de $fluide_2$
6. Pour la demande 2, on va chercher dans le magasin 3 1 unité de $fluide_2$

Ce qui donne les tableaux itérés suivants:

	F1	**F2**
M1	0.5	1
M2	1	2
M3	2	1

	F1	**F2**
D1	0	0
D2	1	3

1.

	F1	**F2**
M1	0	1
M2	1	2
M3	2	1

	F1	**F2**
D1	0	0
D2	0.5	3

2.

	F1	**F2**
M1	0	0
M2	1	2
M3	2	1

	F1	**F2**
D1	0	0
D2	0.5	3

3.

	F1	**F2**
M1	0	0
M2	0.5	2
M3	2	1

	F1	**F2**
D1	0	0
D2	0	3

4.

	F1	**F2**
M1	0	0
M2	0.5	0
M3	2	1

	F1	**F2**
D1	0	0
D2	0	1

5.

	F1	**F2**
M1	0	0
M2	0.5	0
M3	2	0

	F1	**F2**
D1	0	0
D2	0	0

6.

Cela montre que la solution est cohérente étant donné que chaque état de la solution respecte les contraintes données.

Cas particulier 1.2

Variables

Les variables du problème sont les suivantes:

- q_{mfd} : Quantité de fluide f stockée dans le magasin m pour la demande d.
- b_{md} : Variable binaire indiquant s'il y une commande au magasin m pour la demande d.

Fonction Objectif

Le problème vise à minimiser le coût total, défini par la fonction objectif suivante :

$$min \sum_{m=1}^{nbM} \sum_{f=1}^{nbF} \sum_{d=1}^{nbD} q_{mfd} \cdot (COUTS_{mf} + COUTSFIXES_{dm}) + \sum_{m=1}^{nbM} \sum_{d=1}^{nbD} b_{md} \cdot COUTSVAR_{dm}$$

Contraintes

Les contraintes du problème sont les suivantes :

- Demande respectée : $\forall d \in [1, \ldots, nbD], \forall f \in [1, \ldots, nbF], \text{DEMANDES}_{df} \leq \sum_{m=1}^{nbM} q_{mfd}$
- Stock respecté : $\forall m \in [1, \ldots, nbM], \forall f \in [1, \ldots, nbF], \sum_{d=1}^{nbD} q_{mfd} \leq \text{STOCK}_{mf}$
- Contrainte de disponibilité : $\forall m \in [1, \ldots, nbM], \forall d \in [1, \ldots, nbD], \sum_{f=1}^{nbF} q_{mfd} \leq M \cdot b_{md}$

Domaine

Les domaines des variables sont les suivants :

- $$\forall m \in [1, \ldots, nbM], \forall f \in [1, \ldots, nbF], \forall d \in [1, \ldots, nbD], \begin{cases} q_{mfd} \geq 0 \\ b_{md} \in \{0, 1\} \end{cases}$$

Conclusion

Avec les données de l'énoncé nous obtenons le résultat suivant :

```
Problem:    ECommerceCP12
Rows:       17
Columns:    18 (6 integer, 6 binary)
Non-zeros:  60
Status:     INTEGER OPTIMAL
Objective:  coutTotal = 621 (MINimum)

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 coutTotal                 621
     2 demandeRespectee[1,1]
                                  -2                          -2
     3 demandeRespectee[1,2]
                                   0                          -0
     4 demandeRespectee[2,1]
                                  -1                          -1
     5 demandeRespectee[2,2]
                                  -3                          -3
     6 stockRespecte[1,1]
                                   0                         2.5
     7 stockRespecte[1,2]
                                   0                           1
     8 stockRespecte[2,1]
                                   1                           1
     9 stockRespecte[2,2]
                                   2                           2
    10 stockRespecte[3,1]
                                   2                           2
    11 stockRespecte[3,2]
                                   1                           1
    12 bDef[1,1]                   0                          -0
    13 bDef[1,2]                   0                          -0
    14 bDef[2,1]                   0                          -0
    15 bDef[2,2]                -6.5                          -0
    16 bDef[3,1]                -7.5                          -0
    17 bDef[3,2]                -8.5                          -0

   No. Column name       Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 q[1,1,1]                    0             0
     2 q[1,2,1]                    0             0
     3 q[1,1,2]                    0             0
     4 q[1,2,2]                    0             0
     5 q[2,1,1]                    0             0
     6 q[2,2,1]                    0             0
     7 q[2,1,2]                    1             0
     8 q[2,2,2]                    2             0
     9 q[3,1,1]                    2             0
```

```
    10 q[3,2,1]                    0             0
    11 q[3,1,2]                    0             0
    12 q[3,2,2]                    1             0
    13 b[1,1]       *              0             0             1
    14 b[1,2]       *              0             0             1
    15 b[2,1]       *              0             0             1
    16 b[2,2]       *              1             0             1
    17 b[3,1]       *              1             0             1
    18 b[3,2]       *              1             0             1

Integer feasibility conditions:

KKT.PE: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

KKT.PB: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

End of output
```

On pourrait refaire le même procédé itératif avec les tableaux pour montrer la faisabilité de la solution.
Cependant l'affichage de 4 tableaux cote-à-cote pose des problèmes de lisibilité.
Ici on remarque juste que pour les mêmes demandes que pour l'exercice précédent les réponses ne sont pas les mêmes.
Ce qui prouve que les contraintes de coûts fixes et variables ont été prise en compte par GLPK.

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}$$

Cas particulier 2

Ce problème correspond au problème théorique du voyageur car il s'agit de trouver le chemin le plus court pour livrer les colis au bon endroit.

Variables

Dans ce problème nous avons identifié 2 variables :

- Y : Matrice des arcs entre les clients. Le coefficient $c(i, j)$ indique dans le cas où il vaut 1 que le livreur part du client i et se dirige vers le client j
- T : Vecteur de l'ordre de livraison

Fonction Objectif

La fonction à minimiser dans le problème est la fonction :

$$min \sum_{i=1}^{n} \sum_{j=1}^{n} DISTANCE_{ij} * Y_{ij}$$

Contraintes

Assure qu'il y ait un seul et unique 1 par ligne:

$$\sum_{j=1}^{N} Y[i,j] = 1, \quad \forall i \in \{1, 2, \ldots, N\}$$

Assure qu'il y ait un seul et unique 1 par colonne:

$$\sum_{i=1}^{N} Y[i,j] = 1, \quad \forall j \in \{1, 2, \ldots, N\}$$

Assure qu'un client ne peut pas se visiter après s'être visité :

$$Y[i,i] = 0, \quad \forall i \in \{1, 2, \ldots, N\}$$

Assure qu'il n'y ait pas de sous-cycle au sein du graphe representant les clients et leur lirvraison:

$$T[j] + (1 - Y[i,j]) \cdot M \geq T[i] + 1, \quad \forall i \in \{1, 2, \ldots, N\}, \forall j \in \{2, 3, \ldots, N\}$$

Assure que l'ordre est positif:

$$T[i] \geq 0, \quad \forall i \in \{1, 2, \ldots, N\}$$

Domaine

- $Y \in \mathcal{M}_n(\mathbb{B})$
- $T \in \mathbb{R}^n$

Conclusion

Avec les paramètres précedent nous obtenons la solution suivante avec GLPK :

```
Problem:    ECommerceCP2
Rows:       55
Columns:    42 (42 integer, 36 binary)
Non-zeros:  200
Status:     INTEGER OPTIMAL
Objective:  Distance = 22 (MINimum)

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 Distance                   22
     2 RegleUn[1]                  1             1             =
     3 RegleUn[2]                  1             1             =
     4 RegleUn[3]                  1             1             =
     5 RegleUn[4]                  1             1             =
     6 RegleUn[5]                  1             1             =
     7 RegleUn[6]                  1             1             =
     8 RegleDeux[1]                1             1             =
     9 RegleDeux[2]                1             1             =
    10 RegleDeux[3]                1             1             =
    11 RegleDeux[4]                1             1             =
    12 RegleDeux[5]                1             1             =
    13 RegleDeux[6]                1             1             =
    14 RegleTrois[1]
                                   0            -0             =
    15 RegleTrois[2]
                                   0            -0             =
    16 RegleTrois[3]
                                   0            -0             =
    17 RegleTrois[4]
                                   0            -0             =
    18 RegleTrois[5]
                                   0            -0             =
    19 RegleTrois[6]
                                   0            -0             =
    20 RegleQuatre[1,2]
                                -999          -999
    21 RegleQuatre[1,3]
                                   5          -999
    22 RegleQuatre[1,4]
                                   2          -999
    23 RegleQuatre[1,5]
```

```
                                        3            -999
    24 RegleQuatre[1,6]
                                        4            -999
    25 RegleQuatre[2,2]
                                        0            -999
    26 RegleQuatre[2,3]
                                        4            -999
    27 RegleQuatre[2,4]
                                     -999            -999
    28 RegleQuatre[2,5]
                                        2            -999
    29 RegleQuatre[2,6]
                                        3            -999
    30 RegleQuatre[3,2]
                                       -4            -999
    31 RegleQuatre[3,3]
                                        0            -999
    32 RegleQuatre[3,4]
                                       -3            -999
    33 RegleQuatre[3,5]
                                       -2            -999
    34 RegleQuatre[3,6]
                                       -1            -999
    35 RegleQuatre[4,2]
                                       -1            -999
    36 RegleQuatre[4,3]
                                        3            -999
    37 RegleQuatre[4,4]
                                        0            -999
    38 RegleQuatre[4,5]
                                     -999            -999
    39 RegleQuatre[4,6]
                                        2            -999
    40 RegleQuatre[5,2]
                                       -2            -999
    41 RegleQuatre[5,3]
                                        2            -999
    42 RegleQuatre[5,4]
                                       -1            -999
    43 RegleQuatre[5,5]
                                        0            -999
    44 RegleQuatre[5,6]
                                     -999            -999
    45 RegleQuatre[6,2]
                                       -3            -999
    46 RegleQuatre[6,3]
                                     -999            -999
    47 RegleQuatre[6,4]
                                       -2            -999
    48 RegleQuatre[6,5]
                                       -1            -999
```

```
    49 RegleQuatre[6,6]
                                   0          -999
    50 Reglex[1]                   0            -0
    51 Reglex[2]                   1            -0
    52 Reglex[3]                   5            -0
    53 Reglex[4]                   2            -0
    54 Reglex[5]                   3            -0
    55 Reglex[6]                   4            -0

   No. Column name       Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 Y[1,1]       *              0             0             1
     2 Y[1,2]       *              1             0             1
     3 Y[1,3]       *              0             0             1
     4 Y[1,4]       *              0             0             1
     5 Y[1,5]       *              0             0             1
     6 Y[1,6]       *              0             0             1
     7 Y[2,1]       *              0             0             1
     8 Y[2,2]       *              0             0             1
     9 Y[2,3]       *              0             0             1
    10 Y[2,4]       *              1             0             1
    11 Y[2,5]       *              0             0             1
    12 Y[2,6]       *              0             0             1
    13 Y[3,1]       *              1             0             1
    14 Y[3,2]       *              0             0             1
    15 Y[3,3]       *              0             0             1
    16 Y[3,4]       *              0             0             1
    17 Y[3,5]       *              0             0             1
    18 Y[3,6]       *              0             0             1
    19 Y[4,1]       *              0             0             1
    20 Y[4,2]       *              0             0             1
    21 Y[4,3]       *              0             0             1
    22 Y[4,4]       *              0             0             1
    23 Y[4,5]       *              1             0             1
    24 Y[4,6]       *              0             0             1
    25 Y[5,1]       *              0             0             1
    26 Y[5,2]       *              0             0             1
    27 Y[5,3]       *              0             0             1
    28 Y[5,4]       *              0             0             1
    29 Y[5,5]       *              0             0             1
    30 Y[5,6]       *              1             0             1
    31 Y[6,1]       *              0             0             1
    32 Y[6,2]       *              0             0             1
    33 Y[6,3]       *              1             0             1
    34 Y[6,4]       *              0             0             1
    35 Y[6,5]       *              0             0             1
    36 Y[6,6]       *              0             0             1
    37 T[2]         *              1
    38 T[1]         *              0
    39 T[3]         *              5
    40 T[4]         *              2
```

```
    41 T[5]         *              3
    42 T[6]         *              4

Integer feasibility conditions:

KKT.PE: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

KKT.PB: max.abs.err = 0.00e+00 on row 0
        max.rel.err = 0.00e+00 on row 0
        High quality

End of output
```

Vérifions que cette solution est cohérente. La matrice Y nous indique que le livreur emprunte le chemin suivant :

$$ALPHA \rightarrow client1 \rightarrow client3 \rightarrow client4 \rightarrow client5 \rightarrow ALPHA$$

Cette solution est cohérente pour les raisons suivantes :

- En terme de graphe c'est un cycle hamiltonien. On passe une et une seule fois chez chaque client.
- Chaque ligne et colonne ne possède qu'un seul 1 et le reste de 0.