RAPPORT DE TP
RECHERCHE OPÉRATIONNELLE
TP1

2SN-L4
OUDARD VERON
TENE FOGANG ZACHARIE IGOR
Abdelmounaim SALOUANI

TABLE DE MATIERE

1. Modélisation du problème d'assemblage

- Choix des variables

C : Nombre de vélos cargos produits par semaine.
S : Nombre de vélos standards produits par semaine.

- Définition de la fonction objectif

L'objectif étant de maximiser la marge totale alors :

$$\mathbf{Z = 700 \cdot C + 300 \cdot S}$$

- Définition des contraintes

L'énoncé impose plusieurs contraires qui sont:

1. Contrainte sur le temps de travail disponible :

$$6 \cdot C + 5 \cdot S \leq 6000$$

2. Contrainte sur l'espace de stockage disponible :

$$2.5 \cdot C + S \leq 1500 \iff (5 \cdot C + 2 \cdot S \leq 3000)$$

3. Contrainte sur la capacité de production de vélo :

$$C \leq 700$$

4. Contrainte naturelle (positivité) : $C \geq 0, \quad S \geq 0$

- Cas de la programmation linéaire (PL)

$$\mathbf{Max\ Z = 700 \cdot C + 300 \cdot S}$$

Sous les contraintes :

$$6 \cdot C + 5 \cdot S \leq 6000$$
$$5 \cdot C + 2 \cdot S \leq 1500$$
$$C \leq 700$$
$$C , S \geq 0$$

Dans ce cas, les variables C et S sont continues. $C,S \in \mathbb{R}$.

- Cas de la programmation linéaire en nombre entier (PLNE)

  Il s'agit du même système que celui de programmation linéaire établi précédemment. Seul le domaine des variables changent. En effet, ici , les variables sont discrètes.

$$\mathbf{Max\ Z = 700 \cdot C + 300 \cdot S}$$

Sous les contraintes :

$$6 \cdot C + 5 \cdot S \leq 6000$$
$$5 \cdot C + 2 \cdot S \leq 1500$$
$$C \leq 700$$
$$C , S \geq 0$$
$$C , S \in N$$

Le problème ainsi posé permet d'éviter la fabrication de vélos partiellement montés. En effet, la résolution de ce problème permet de trouver les solutions pour lesquelles on aura pas de restes de vélos fabriqués.

- Résolution par le solveur

Le problème est assez simple et nous n'utilisons que 2 variables scalaires, par conséquent, nous avons choisis de travailler avec le format **"lp".**

Dans le cas continu (PL), on obtient un bénéfice maximal de **438 461.5385 €**, pour les variables C = **230.769** et S = **923.077** .
Dans le cas de la programmation linéaire en nombre entier, on obtient un bénéfice maximal de **438 400 €**, pour les variables C = **232** et S = **920**. Les résultats obtenus sont similaires à ceux du cas continu, ce qui est cohérent pour ce problème.

Les résultats obtenus par le solveur sont présentés sur les figure 1 et 2 ci-dessous.

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

   No.   Row name   St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- ------------- -------------
     1 TEMPS_DE_TRAVAIL
                    NU          6000                        6000       7.69231
     2 ESPACE_DE_STOCKAGE
                    NU          3000                        3000       130.769
     3 LIMITE_CARGOS
                    B        230.769                         700

   No. Column name  St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- ------------- -------------
     1 C            B        230.769             0
     2 S            B        923.077             0
```

figure 1 :   cas de la programmation linéaire

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

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 TEMPS_DE_TRAVAIL
                                5992                        6000
     2 ESPACE_DE_STOCKAGE
                                3000                        3000
     3 LIMITE_CARGOS
                                 232                         700

   No. Column name       Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 C            *            232             0
     2 S            *            920             0
```

figure 2 : cas de la programmation linéaire en nombre entier

2. Modélisation du problème d'attribution des tâches avec prise en compte des préférences (PLNE)

Nous notons C la matrice associée aux scores personne/tâche. Il y' a N tâches C(i,j) correspondant au score de préférence la personne Pi pour la tâche Tj.

- Choix des variables

  X : matrice binaire d'association personne/tâche.
  Xij = 1 si la personne i est associé à la tâche j,
  0 sinon.
  Ainsi, X ∈ {0,1}NxN . (Xij variable binaire)

- Définition de la fonction objectif

  La manageuse souhaite **maximiser** le score total des préférences de l'équipe pour les tâches assignées. Le score de préférence pour Pi et Tj étant Cij, la fonction objectif est donc :

$$Max \sum_{\substack{1 < i < N \\ 1 < j < N}} C(i,j)X(i,j)$$

- Définition des contraintes

1. Chaque personne doit effectuer exactement une tâche c'est à dire la somme des éléments d'une ligne doit être égale à 1

$$\forall i, \sum_{J=1}^{N} X(i,j) = 1$$

2. Chaque tâche doit être effectuée par exactement une personne c'est à dire la somme des éléments d'une colonne doit être égale à 1

$$\forall j, \sum_{i=1}^{N} X(i,j) = 1$$

3. Les variables de décisions sont binaires.

- Résolution par le solveur

Nous avons utilisé un fichier **.mod** et un fichier **.dat** car notre variable X et le paramètre C (matrice des scores de préférence) dépendent du paramètre n (nombre de personnes et de tâches). Ce problème nécessitant la manipulation de structures complexes (les matrices),il nous été judicieux d'utiliser le format **gmpl** qui permettait de plus facilement manipuler les matrices.

Pour tester notre modèle, nous avons utilisé 03 exemples (exemples fournis dans les fichiers *pbtache.dat.txt, pbtache2.dat.txt,).*

Exemple 1 : La matrice de préférence/score que nous avons utilisé est la suivante:

	T1	T2	T3 :=
P1	4	7	9
P2	9	8	3
P3	2	1	2;

A la main, le score maximal que l'on peut obtenir conformément au problème posé est **z=19** et la répartition optimale serait : **(P1,T3), (P2,T1) et (P3,T1)**. Les résultats obtenus après résolution par le solveur sont présentés à la figure 3.

Exemple 2 : Nous avons utilisé :

	T1	T2	T3 :=
P1	9	7	0
P2	4	1	9
P3	2	8	5;

Comme précédemment, une résolution à la main est évidente.On trouve **z=26** comme score maximal et la répartition optimale serait: **(P1,T1),(P2,T3) et (P3,T2)**. Les résultats obtenus après résolution par le solveur sont présentés à la figure 4.

```
Problem:    pbtache
Rows:       7
Columns:    9 (9 integer, 9 binary)
Non-zeros:  27
Status:     INTEGER OPTIMAL
Objective:  TotalPreferences = 19 (MAXimum)

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 TotalPreferences
                                  19
     2 UneTacheParPersonne[P1]
                                   1             1             =
     3 UneTacheParPersonne[P2]
                                   1             1             =
     4 UneTacheParPersonne[P3]
                                   1             1             =
     5 UnePersonneParTache[T1]
                                   1             1             =
     6 UnePersonneParTache[T2]
                                   1             1             =
     7 UnePersonneParTache[T3]
                                   1             1             =

   No. Column name       Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 x[P1,T1]     *              0             0             1
     2 x[P1,T2]     *              0             0             1
     3 x[P1,T3]     *              1             0             1
     4 x[P2,T1]     *              1             0             1
     5 x[P2,T2]     *              0             0             1
     6 x[P2,T3]     *              0             0             1
     7 x[P3,T1]     *              0             0             1
     8 x[P3,T2]     *              1             0             1
     9 x[P3,T3]     *              0             0             1
```

figure 3 :   résultats obtenus (pbtache.dat.txt)

```
Problem:    pbtache
Rows:       7
Columns:    9 (9 integer, 9 binary)
Non-zeros:  26
Status:     INTEGER OPTIMAL
Objective:  TotalPreferences = 26 (MAXimum)

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 TotalPreferences
                              26
     2 UneTacheParPersonne[P1]
                               1             1             =
     3 UneTacheParPersonne[P2]
                               1             1             =
     4 UneTacheParPersonne[P3]
                               1             1             =
     5 UnePersonneParTache[T1]
                               1             1             =
     6 UnePersonneParTache[T2]
                               1             1             =
     7 UnePersonneParTache[T3]
                               1             1             =

   No. Column name       Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 x[P1,T1]     *              1             0             1
     2 x[P1,T2]     *              0             0             1
     3 x[P1,T3]     *              0             0             1
     4 x[P2,T1]     *              0             0             1
     5 x[P2,T2]     *              0             0             1
     6 x[P2,T3]     *              1             0             1
     7 x[P3,T1]     *              0             0             1
     8 x[P3,T2]     *              1             0             1
     9 x[P3,T3]     *              0             0             1
```

figure 4 : résultats obtenus (pbtache2.dat.txt)

Les résultats sont cohérents, on peut vérifier à la main que les résultats trouvés sont bien les maxima. Les répartition obtenues (la matrice X) est bien conforme aux résultats attendus.

3. Applications en optimisation pour l'e-commerce : Modélisation du problème d'affectation des commandes

Nous notons C la matrice des coûts de livraison des fluides par magasin. C(i,j) désigne le coût de livraison du fluide j depuis le magasin i. S la matrice des stocks de fluides par magasin et D la matrice des commandes de fluides. S(i,j) désigne la quantité (stock) de fluide j disponible en magasin i et D(i,j) représente la quantité de fluide j demandée lors de la commande i.
Nous désignons également par les lettres D ,M, F le nombre de commandes , le nombre de magasins et le nombre de variétés de fluides respectivement.

A. cas particulier 1.1

- Choix des variables

La quantité de fluide j livrée depuis le magasin i pour la commande k que nous noterons :

$$x_{\mathrm{ijk}}$$

- Définition de la fonction objectif

Le but ici est de **minimiser** le coût total de la livraison que nous noterons Z et dont l'expression est définie par :

$$\sum_{i=1}^{M}\sum_{j=1}^{F}\sum_{k=1}^{D} X_{ijk}C_{ij}$$

- Définition des contraintes

1. Respect des demandes (Pour chaque commande k et chaque fluide j, la quantité livrée doit être égale à la demande) :

$$\sum_{i=1}^{M} X_{ijk} = D_{kj}, \forall k \in [1, D], \forall j \in [1, F]$$

2. Respect des stocks (Pour chaque magasin i et chaque fluide j, la quantité totale de fluide livrée ne doit pas dépasser le stock disponible) :

$$\sum_{k=1}^{D} X_{ijk} \leq S_{ij}, \forall i \in [1, M], \forall j \in [1, F]$$

3. Domaine des variables (contraintes naturelles: les quantités livrées doivent être non négatives) :

$$x_{\text{ijk}} \geq 0$$

- Résolution par le solveur

Nous avons utilisé le format **gmpl** compte tenu de la manipulation des matrices dans notre problème.

Pour tester notre modèle, nous avons utilisé l' exemple fourni dans l'énoncé (*exple.dat.txt*) du problème :

	F1	F2
D1	2	0
D2	1	3

(a) Fluides demandés par commande

	F1	F2
M1	2.5	1
M2	1	2
M3	2	1

(b) Stocks de fluides par magasin

	F1	F2
M1	1	1
M2	2	3
M3	3	2

(c) Coûts unitaires par magasin d'origine

Une résolution à la main permet de trouver que le **coût minimal est de 9.5** et que la répartition optimale serait :

Magasin i	Fluide j	Commande k	Quantité x(i,j,k)
M1	F1	D1	2
M1	F1	D2	0.5
M2	F1	D2	0.5
M1	F2	D2	1
M2	F2	D2	1
M3	F2	D2	1

figure 5 : tableau représentant la solution obtenue à la main

```
Problem:    casparticulier1
Rows:       11
Columns:    12
Non-zeros:  36
Status:     OPTIMAL
Objective:  TotalCost = 9.5 (MINimum)

   No.   Row name   St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- ------------- -------------
     1 SatisfaireDemande[D1,F1]
                    NS             2             2             =             2
     2 SatisfaireDemande[D1,F2]
                    B              0            -0             =
     3 SatisfaireDemande[D2,F1]
                    NS             1             1             =             2
     4 SatisfaireDemande[D2,F2]
                    NS             3             3             =             3
     5 RespecterStock[M1,F1]
                    NU           2.5                         2.5            -1
     6 RespecterStock[M1,F2]
                    NU             1                           1            -2
     7 RespecterStock[M2,F1]
                    B            0.5                           1
     8 RespecterStock[M2,F2]
                    B              1                           2
     9 RespecterStock[M3,F1]
                    B              0                           2
    10 RespecterStock[M3,F2]
                    NU             1                           1            -1
    11 TotalCost    B            9.5

   No. Column name  St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- ------------- -------------
     1 x[M1,F1,D1]  B              2             0
     2 x[M2,F1,D1]  NL             0             0                       < eps
     3 x[M3,F1,D1]  NL             0             0                           1
     4 x[M1,F2,D1]  NL             0             0                           3
     5 x[M2,F2,D1]  NL             0             0                           3
     6 x[M3,F2,D1]  NL             0             0                           3
     7 x[M1,F1,D2]  B            0.5             0
     8 x[M2,F1,D2]  B            0.5             0
     9 x[M3,F1,D2]  NL             0             0                           1
    10 x[M1,F2,D2]  B              1             0
    11 x[M2,F2,D2]  B              1             0
    12 x[M3,F2,D2]  B              1             0
```

figure 6 : résultats obtenus exemple donné dans l'énoncé(exple.dat.txt)

Après résolution par le solveur, nous obtenons les **résultats attendus**.

B. cas particulier 1.2

Etant un prolongement du **cas particulier 1.1**, nous suivons une logique similaire à ce dernier. Nous considérons en plus des notations précédemment définies , **C'** la matrice des coûts fixes et **C"**. **C'(i,k)** désigne ainsi le coût fixe d'expédition d'un colis du magasin i au client k et **C"(i,k)** le coût variable par unité transportée du magasin i au client k.

- Choix des variables

  Quantité de fluide j livrée du magasin i au client k : **X(i,j,k)**
  Variable binaire représentant si le magasin i dessert le client k: **Y(i,k)**

- Définition de la fonction objectif

  Le but est de minimiser le coût total d'expédition, qui comprend le **coût fixe** (réalisé une seule fois pour chaque magasin i desservant un client k) et le **coût variable** qui dépend de la quantité transportée.

$$\sum_{i=1}^{M}\sum_{k=1}^{D}\left(C'(i,k)Y(i,k)+\sum_{j=1}^{F}C''(i,k)X(i,j,k)\right)$$

- Définition des contraintes

  1. Respect de la demande (chaque client k doit recevoir la quantité totale de fluide j qu'il demande) :

$$\sum_{i=1}^{M}X_{ijk}=D_{kj},\ \forall k\in[1,D],\ \forall j\in[1,F]$$

  Où D(k,j) ici désigne la quantité demandée de fluide j par le client k.

  2. Respect de stocks (Pour chaque magasin i et chaque fluide j, la quantité totale de fluide livrée ne doit pas dépasser le stock disponible)

$$\sum_{k=1}^{D}X_{ijk}\le S_{ij},\ \forall i\in[1,M],\ \forall j\in[1,F]$$

3. Domaine des variables (contraintes naturelles: les quantités livrées doivent être non négatives) :

$$x_{\text{ijk}} \geq 0$$

4. Binarité des variables Y(i,k)

5.Contrainte de conformité (garantissant que si Y(i,k)=0 alors X(i,j,k)=0)

Nous introduisons la variable **U** défini comme suite :

$$Max \sum_{j=1}^{F} D(k,j), \forall k$$

Cela correspond à la somme maximale des demandes pour tous les fluides j d'un client k qui est suffisamment grand pour la contrainte et lâche .

Nous écrivons donc la **contrainte**:

$$\sum_{j=1}^{F} X(i,j,k) \leq UY(i,k), \forall i \in [1,M], \forall k \in [1,D]$$

- Résolution par le solveur

Nous avons utilisé le format **gmpl** compte tenu de la manipulation des matrices dans notre problème.

Pour tester notre modèle, nous avons utilisé l' exemple fourni dans l'énoncé (*exple2.dat.txt*) du problème :

	M1	M2	M3
D1	110	90	100
D2	110	90	100

(d) Coûts fixes d'expédition d'un colis entre chaque paire : point de demande, magasin

	M1	M2	M3
D1	10	1	5
D2	2	20	10

(e) Coûts variables d'expédition d'un colis entre chaque paire : point de demande, magasin

```
Problem:    casparticulier1
Rows:       23
Columns:    18 (6 integer, 6 binary)
Non-zeros:  66
Status:     INTEGER OPTIMAL
Objective:  TotalCost = 354 (MINimum)

   No.  Column name       Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 x[M1,F1,D1]                 0             0
     2 x[M2,F1,D1]                 0             0
     3 x[M3,F1,D1]                 2             0
     4 x[M1,F2,D1]                 0             0
     5 x[M2,F2,D1]                 0             0
     6 x[M3,F2,D1]                 0             0
     7 x[M1,F1,D2]                 1             0
     8 x[M2,F1,D2]                 0             0
     9 x[M3,F1,D2]                 0             0
    10 x[M1,F2,D2]                 1             0
    11 x[M2,F2,D2]                 2             0
    12 x[M3,F2,D2]                 0             0
    13 y[M1,D1]     *              0             0             1
    14 y[M1,D2]     *              1             0             1
    15 y[M2,D1]     *              0             0             1
    16 y[M2,D2]     *              1             0             1
    17 y[M3,D1]     *              1             0             1
    18 y[M3,D2]     *              0             0             1
```

figure 7 : résultats obtenus exemple donné dans l'énoncé(exple2.dat.txt)

Après résolution par le solveur, le **coût minimal** obtenu est de **354**.

C. cas particulier 2

Ce cas particulier correspond à un problème d'optimisation des **tournées de livraison**, il rappel le problème du **problème du voyageur de commerce** vu en cours mais adapté au contexte des livraison

- Choix des variables

X(i,j) : Variable binaire qui vaut 1 si le livreur se déplace du client i au client j, et 0 sinon.

Ti : variable ordre de visite du client (on la définit pour assurer la continuité de la tournée).

- Définition de la fonction objectif

Le but est de minimiser la distance totale parcourue d'où :

$$Min\sum_{i=1}^{n}\sum_{j=1}^{n} distance(i,j)X(i,j)$$

- Définition des contraintes

1. Chaque client est visité une et une seule fois : X(i,i)=0 et

$$\sum_{j=1}^{n} X(i,j) = 1 \qquad \sum_{i=1}^{n} X(i,j) = 1$$

2. Une contrainte supplémentaire pour éviter des sous-tours :

$$Ti - Tj + nX(i,j) \leq n - 1, \forall i, j$$

3. Domaine des variables : X(i,j) dans {0,1} et Ti positif.

- Résolution par le solveur

Pour les mêmes raisons évoquées pour les problèmes précédents, nous avons utilisé le format **gmpl** (compte tenu de la manipulation des matrices dans notre problème). Pour tester notre modèle, nous avons utilisé l' exemple fourni dans l'énoncé (*exple3.dat.txt*) du problème :

```
Problem:    casparticulier1
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             =
```

figure 8 : résultats obtenus exemple donné dans l'énoncé(exple3.dat.txt)

Après résolution par le solveur, la **distance totale parcourue** est de 22.