Rapport TP1 - Recherche Operationnele

AKKAR KHADIJA, ABOUMEJD WISSAL
L1 - 2023/2024

Table des matières

1 Résolution de PL/PLNE avec le solveur

1.1 Assemblage

L'objectif de cette partie est de déterminer la répartition optimale du travail entre l'assemblage de vélos cargos (C) et de vélos standards (S) dans l'usine, en tenant compte des contraintes de temps et d'espace de stockages et en visant à maximiser la marge totale générée par la vente des vélos assemblés.

1. Sets :
Le set VELOS a été défini pour représenter les deux types de vélos à assembler : cargos (C) et standards (S).

2. Variables :
La variable Q[i] a été introduite pour chaque type de vélo i dans l'ensemble VELOS, représentant la quantité à assembler de chaque type.

3. Constantes :
-surface[i] : la surface occupée par un vélo de type i dans le parking.

-dureefab[i] : la durée nécessaire pour assembler un vélo de type i.

-marge[i] : la marge financière pour un vélo de type i.

-maxVelos[i] : la quantité maximale de vélos de type i pouvant être assemblée par semaine.

4. Contraintes :
-Respect de l'Espace Maximal : garantit que la somme des surfaces occupées par les vélos assemblés ne dépasse pas la capacité du parking (1500m2).

-Contrainte de Temps : assure que la somme des temps d'assemblage ne dépasse pas le temps de travail maximal disponible (6000 heures par semaine).

-Contrainte de Quantité : fixe une limite sur la quantité maximale de chaque type de vélo assemblée par semaine.

5. Résultat :

Modèle PL : La solution du modèle PL indique que la marge totale maximale obtenue est de 438,461.54. Les quantités optimales d'assemblage sont les suivantes :

Q[C] (vélos cargos) : 1500 unités
Q[S] (vélos standards) : 6000 unités

Modèle PLNE : La solution du modèle PLNE, qui tient compte des contraintes de quantité en nombres entiers, montre une marge totale maximale de 438,400. Les quantités optimales d'assemblage sont :

Q[C] (vélos cargos) : 1500 unités
Q[S] (vélos standards) : 5992 unités

Les deux modèles ont abouti à des solutions optimales, mais le modèle PLNE a imposé des contraintes supplémentaires pour les quantités de vélos standards, les limitant à 5992 unités au lieu de 6000. Cette restriction a un impact sur la marge totale, qui est légèrement inférieure dans le modèle PLNE.

1.2 Affectation avec prise en compte des préférences

Une manageuse doit établir un planning pour son équipe composée de N personnes, chaque personne devant effectuer exactement une tâche parmi les N tâches disponibles. Pour formaliser ce problème, nous pouvons définir un modèle (PLNE). Chaque personne i doit être affectée à une tâche j de manière à maximiser le score total des préférences.

1. Sets :
P, représentant l'ensemble des personnes.
T, représentant l'ensemble des tâches à effectuer.

2. Variables :
La variable X[i,j], définie comme binaire, indique si la personne i est assignée à la tâche j. Cette variable permet de modéliser l'affectation des tâches de manière efficace..

3. Constantes :
La constante c[i,j] représente le score de préférence de la personne i pour la tâche j. Ces scores sont utilisés pour calculer la fonction objectif.

4. Contraintes :
Assignation Personnelle : La contrainte "AssignPerson" garantit qu'une personne est assignée exactement à une seule tâche.

$$\sum_{j=1}^{N} x_{ij} = 1 \quad \text{pour tout } i.$$

Assignation de Tâche : La contrainte "AssignTask" garantit qu'une tâche est effectuée exactement une fois.

$$\sum_{i=1}^{N} x_{ij} = 1 \quad \text{pour tout } j.$$

Les variables d'affectation sont binaires :

$$x_{ij} \in \{0, 1\}.$$

5. Objectif :
Maximiser la somme pondérée des scores de préférence en fonction des affectations, représenté par la fonction objectif "MaximizePreference" :

$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} \cdot x_{ij}$$

6. Résultat :
Pour l'ensemble de données suivant :

```
data;

set P := personne1 personne2 personne3;
set T := tache1 tache2 tache3;

param c :=
  personne1 tache1 8
  personne1 tache2 5
  personne1 tache3 6
  personne2 tache1 7
  personne2 tache2 9
  personne2 tache3 4
  personne3 tache1 5
  personne3 tache2 6
  personne3 tache3 7;

end;
```

FIGURE. 1 – Exemple pour notre problème

La fonction "MaximizePreference" a atteint la valeur maximale de 24, indiquant une affectation qui maximise la satisfaction globale de l'équipe en tenant compte des préférences individuelles.

2 Applications en optimisation pour l'e-commerce

2.1 Cas particulier 1.1

2.1.1 Cas continu

L'objectif sous-jacent à cette problématique est d'optimiser la gestion des fluides en prélevant de manière efficiente dans le stock de trois magasins afin de répondre de façon optimale aux demandes des clients.

Les données pertinentes incluent des paramètres invariables, mais dont les valeurs sont connues.

- Le nombre de fluides disponibles ou demandés par commande pour chaque magasin ;
- Les demandes spécifiées dans le tableau (a) ;
- Le nombre total de magasins disponibles pour la gestion du stock ;
- Les demandes de fluides par commande (a) ;
- Les niveaux actuels de stock de fluides par magasin(b) ;
- Les coûts unitaires par fluide pour chaque magasin (c).

	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

FIGURE. 2 – Exemple pour notre problème

La variable de décision recherchée est la quantité optimale de fluides à prélever dans chaque magasin pour satisfaire les demandes. Notre approche de modélisation intègre deux contraintes préliminaires qui restreignent les choix des valeurs de décision :

- **RespectQuantiteMax** : Cette première contrainte vérifie que les quantités prélevées pour chaque fluide sont inférieures ou égales aux quantités disponibles dans les magasins respectifs ;
- **BonneAffectationDM** : Cette deuxième contrainte assure que la somme des quantités prélevées pour un fluide, à partir de l'ensemble des magasins, est égale à la quantité demandée de ce fluide dans l'ensemble des commandes.

Les résultats (fluidesSol.sol.txt) obtenus grâce à cette modélisation pour l'instance spécifiée dans le sujet indiquent un bénéfice de 9.5.

2.1.2 Cas discret

Cet exemple représente une situation similaire à l'exemple précédent, avec la seule distinction que ce problème constitue la version discrète de la version antérieure. On a travaillé sur les colis au lieu des fluides

La variable de décision que l'on cherche à déterminer est identique à celle du cas particulier 1.1, à la différence que nous avons spécifié que cette variable est de type entier (integer).

Les résultats (colisSol.sol.txt) obtenus grâce à cette modélisation indiquent un bénéfice de 10.

2.2 Cas particulier 1.2

Dans cette section, les coûts d'expédition des colis des magasins aux clients ont été intégrés. Ainsi, une nouvelle donnée a été introduite sous forme d'un tableau contenant ces coûts.

	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

FIGURE. 3 – Coûts fixes/variables d'expédition d'un colis

Une deuxième variable "Trajet" a été ajoutée, accompagnée de contraintes (cheminColisChoisi/cheminColisNonChoisi) supplémentaires, dans le but de satisfaire la fonction objectif.

Nous avons aussi ajouté un paramètre à la variable quantité= "Q", désigné pour les demandes. Ainsi, nous introduisons une nouvelle variable Q qui est liée aux trois informations suivantes : le magasin, le colis, et la demande du client. Cela a été réalisé dans le but d'adapter la fonction objectif aux différents coûts ajoutés.

Les résultats obtenus grâce à cette modélisation pour l'instance spécifiée dans le sujet indiquent un bénéfice de 368.

2.3 Cas particulier 2

L'objectif consiste à formaliser un problème visant à minimiser les itinéraires de livraison pour les livreurs des magasins.

Le cas décrit correspond au Problème du Voyageur de Commerce (PVC), où le magasin ALPHA charge un livreur de minimiser la distance totale en visitant chaque client une seule fois et en retournant au magasin de départ. Le PVC vise l'optimisation du trajet pour toutes les livraisons.

	ALPHA	C1	C2	C3	C4	C5
ALPHA	-	1	1	10	12	12
C1	1	-	1	8	10	11
C2	1	1	-	8	11	10
C3	10	8	8	-	1	1
C4	12	10	11	1	-	1
C5	12	11	10	1	1	-

FIGURE. 4 – matrice des distances (magasin ALPHA et 5 clients à livrer)

Les données pertinentes sont des paramètres que nous ne pouvons pas contrôler, mais dont nous connaissons les valeurs.

— Le nombre de clients "TAILLE";
— Les clients présentés pour les livraisons "CLIENTS";
— es valeurs des arcs allant de i vers j, représentant les distances à parcourir ou les temps de trajet entre les sites i et j.

La variable de décision que nous cherchons à déterminer est la distance totale parcourue pour une livraison donnée. Donc, on a utilisé la variable binaire "Voie" qui détermine s'il y a une livraison ou non de i à j, puis la variable "Vreste" qui désigne le nombre de livraisons restantes après une.

Nous avons opté pour la modélisation de ce problème en utilisant plusieurs contraintes évoquées dans le sujet, restreignant ainsi les choix des valeurs de décision. Actuellement, nous obtenons un résultat avec un bénéfice de 22 en utilisant le fichier livreaurSol.sol.txt.

2.4 Minimisation des émissions polluantes

Notre objectif est de minimiser les émissions polluantes associées à la livraison de produits depuis des magasins vers des clients. Il est important de noter que notre modèle produit actuellement une valeur minimale nulle. Bien que cela est illogique, nous allons néanmoins exposer notre raisonnement. Malgré nos efforts, nous n'avons pas encore identifié l'erreur responsable de cette valeur minimale nulle.

1. Sets :

Commands : Ensemble des commandes.

Products : Ensemble des produits.

Stores : Ensemble des magasins.

Nodes : Ensemble des nœuds représentant les sites (magasins et commandes).

Arcs : Ensemble des arcs entre les noeuds (représentant les liaisons entre les sites).

2. Variables :

QuantityToDeliver : Variable représentant la quantité de chaque produit à livrer de chaque magasin à chaque commande.

DeliveryRoute : Variable binaire indiquant si une commande est attribuée à un magasin donné.

3. Constantes :
Distance : Paramètre représentant la distance ou le temps de trajet entre chaque paire de sites.

ProductQuantity : Paramètre représentant la quantité de chaque produit dans chaque commande.

StockQuantity : Paramètre représentant le stock de chaque produit dans chaque magasin.

A, B, C : Caractéristiques des véhicules de transport.

4. Contraintes :
-Aucun magasin ne peut livrer plus de produits qu'il n'en a en stock.
-Chaque commande doit être attribuée à un magasin.

En ajoutant toute autre contrainte, on ne trouve pas de résultat.

5. Résultat :
Avec ces données, nous avons trouvé une valeur minimale de pollution nulle. Cela est évidemment illogique, mais nous n'arrivons pas à détecter l'erreur.

3 Conclusion

Dans ce tp nous avons exploré divers problèmes d'optimisation à travers des modèles mathématiques. L'approche globale démontre l'efficacité des modèles pour résoudre des problèmes complexes liés à la gestion de stocks, à l'affectation de tâches, et à l'optimisation des itinéraires de livraison. Ces outils offrent des solutions potentielles pour améliorer l'efficacité opérationnelle dans différents domaines.