Rapport du projet de recherche opérationelle :

Mohammed NAHI(L4)/Achraf KHAIROUN(L3)

Département Sciences du Numérique - Deuxiéme année
2023-2024

Table des matières

1 Assemblage

Le problème concerne l'optimisation de la production dans une usine de vélos qui assemble deux types de vélos : les vélos cargos (C) et les vélos standards (S). L'objectif est de maximiser la marge totale en tenant compte de diverses contraintes opérationnelles.

Choix de Variables de Décision :

1. C : Représente le nombre de vélos cargos à assembler.
2. S : Représente le nombre de vélos standards à assembler.

1.1 Modélisation en PL

Le problème d'assemblage de vélos peut être modélisé comme un problème de programmation linéaire (PL) lorsque les variables de décision, représentant les quantités de vélos de chaque type, peuvent prendre des valeurs continues. Cette modélisation serait appropriée si l'on considère que les quantités de vélos assemblées peuvent prendre des valeurs fractionnaires. Dans ce cas le domaine de Définition serait :

1. C : Contraint à être un nombre réel positif. De plus, C ne peut pas dépasser 700 en raison des limitations liées aux batteries.
2. S : Contraint à être un nombre réel positif.

résultats obtenus :

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

   No.   Row name   St   Activity     Lower bound   Upper bound    Marginal
------ ------------ -- ------------- ------------- ------------- -------------
     1 MaximumDeTravail
                    NU          6000                        6000       7.69231
     2 LimiteDeBatteries
                    B        230.769                         700
     3 LimiteDuParking
                    NU          1500                        1500       261.538
     4 r.13         B        230.769             0
     5 r.14         B        923.077             0

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

FIGURE 1 – Les résultats obtenus pour le problème d'assemblage

Les résultats du modèle sont cohérents avec les contraintes fixées. La production respecte les limites de travail, de batteries et d'espace de stationnement. Les variables de décision sont positives, indiquant une production effective de vélos cargos (C) et standards (S). La marge totale obtenue est de 438461.5385, confirmant l'optimisation réussie du profit.

1.2 Modélisation en PLNE

Le problème d'assemblage peut être modélisé comme un problème de programmation linéaire en nombres entiers (PLNE) lorsque les variables de décision doivent être des nombres entiers, ce qui est souvent le cas dans des contextes pratiques où l'assemblage de vélos ne peut se faire que par unités entières. Ainsi, si l'on veut que les variables de décision représentent des quantités entières de vélos de chaque type, le problème serait formulé comme un PLNE.

Dans ce cas le domaine de Définition serait :

1. C : Contraint à être un nombre entier positif. De plus, C ne peut pas dépasser 700 en raison des limitations liées aux batteries.
2. S : Contraint à être un nombre entier positif.

résultats obtenus :

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

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------- ------------- -------------
     1 MaximumDeTravail
                               5992                          6000
     2 LimiteDeBatteries
                                232                           700
     3 LimiteDuParking
                               1500                          1500

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

FIGURE 2 – Les résultats obtenus pour le problème d'assemblage

Pour le problème d'assemblage de vélos en PLNE, la solution optimale indique la production de 232 vélos de type cargo (C) et 920 vélos de type standard (S). Ces résultats sont cohérents, respectant les contraintes de temps de travail, de capacité de batterie et d'espace de stationnement. La marge totale maximale de 438400 euros a été atteinte, démontrant une allocation efficace des ressources et une réponse adéquate aux exigences opérationnelles. Il est à noter que le profit obtenu est inférieur de 61.5381 euros à celui du cas PL .

2 Affectation avec prise en compte des préférences

L'objectif de ce probléme est d'assigner chaque personne des N personnes à une tâche des N tâches de manière à maximiser le score total des préférences.

Variables : Nous avons choisi la variable binaire Q[i,j] pour modéliser l'affectation des personnes aux tâches. Cette variable prend la valeur 1 si la personne i est assignée à la tâche j, indiquant ainsi l'attribution de la tâche à la personne. En revanche, elle prend la valeur 0 si la

personne n'est pas assignée à la tâche. Cette représentation binaire permet de capturer de manière concise et précise les décisions d'affectation, contribuant ainsi à la formulation du problème d'optimisation d'affectation avec prise en compte des préférences.

Contraintes à respecter :

Pour garantir que chaque personne est attribuée exactement à une tâche une première contrainte est introduite

$$\sum_{i \in \text{PERSONNE}} Q[j,i] = 1 \quad \text{pour chaque tâche } j \in \text{TACHE}.$$

En d'autres termes, la somme des variables Q pour une tâche donnée doit être égale à 1, assurant ainsi qu'une personne est assignée à cette tâche spécifique.

De plus, afin de garantir que chaque travail est attribué, une deuxième contrainte est introduite :

$$\sum_{j \in \text{TACHE}} Q[i,j] = 1 \quad \text{pour chaque personne } i \in \text{PERSONNE}.$$

Cela assure qu'une personne est assignée exactement à une tâche, garantissant ainsi que chaque travail est effectué.

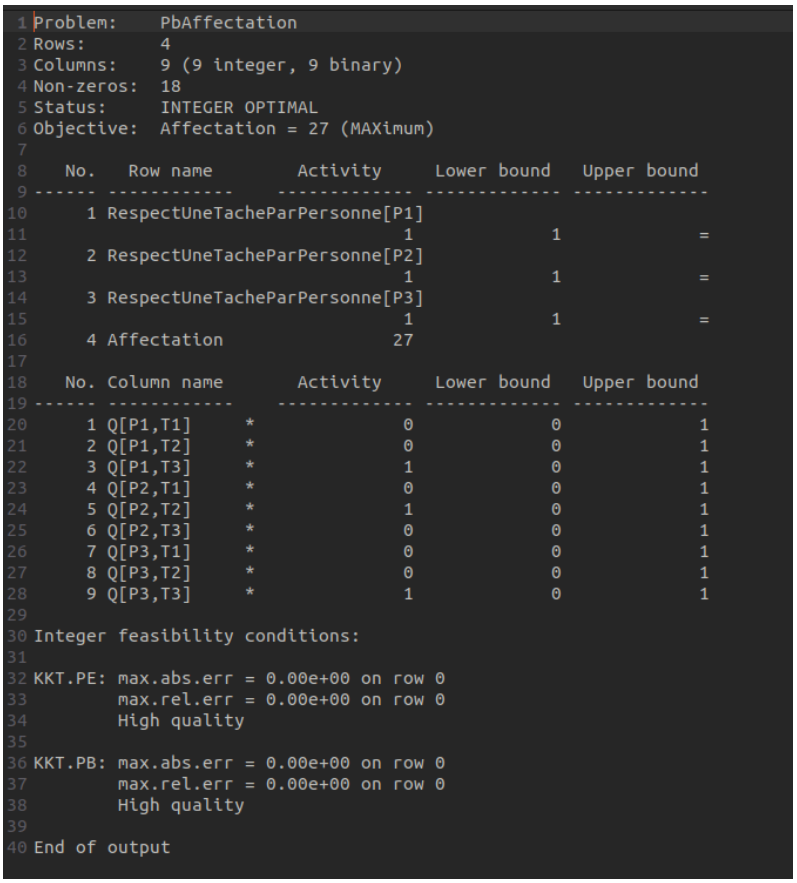
Exemple :

Pour tester le modèle d'affectation avec prise en compte des préférences, nous considérons un ensemble de personnes $(P1, P2, P3)$ et un ensemble de tâches $(T1, T2, T3)$. Les préférences de chaque personne pour chaque tâche sont définies dans la matrice de préférence suivante :

	T1	T2	T3
P1	1	4	9
P2	4	9	5
P3	1	3	9

Ces valeurs représentent les scores de préférence attribués par chaque personne à chaque tâche, notés sur une échelle de 1 à 10. Le modèle d'affectation utilise ces préférences pour optimiser l'attribution des tâches aux personnes, maximisant ainsi la satisfaction globale de l'équipe.

Résultats obtenus :



```
Problem:    PbAffectation
Rows:       4
Columns:    9 (9 integer, 9 binary)
Non-zeros:  18
Status:     INTEGER OPTIMAL
Objective:  Affectation = 27 (MAXimum)

   No.   Row name        Activity     Lower bound   Upper bound
------ ------------    ------------ ------------- -------------
     1 RespectUneTacheParPersonne[P1]
                                1             1             =
     2 RespectUneTacheParPersonne[P2]
                                1             1             =
     3 RespectUneTacheParPersonne[P3]
                                1             1             =
     4 Affectation             27

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

FIGURE 3 – Solution de notre probléme

La solution obtenue pour notre exemple d'affectation est la suivante :

* La personne P1 est assignée à la tâche T1.
* La personne P2 est assignée à la tâche T2.
* La personne P3 est assignée à la tâche T3.

Cela signifie que chaque personne est affectée à exactement une tâche, et chaque tâche est attribuée à exactement une personne, respectant ainsi les contraintes énoncées dans le problème La solution obtenue pour le problème d'affectation est cohérente et satisfait de manière optimale les préférences individuelles de chaque personne. En examinant les scores de préférence attribués dans la matrice, on peut justifier les affectations obtenues.

- La personne $P2$ a attribué le score le plus élevé à la tâche $T2$, ce qui explique pourquoi elle est assignée à $T2$.
- $P1$ et $P3$ ont un score de préférence égal pour la tâche $T3$ (9), mais $P1$ a donné un score de 2 à la tâche $T1$, qui est supérieur au score de $P3$ pour $T1$. Cela explique pourquoi $P1$ prend $T1$ tandis que $P3$ prend $T3$.

En résumé, la solution optimale reflète de manière cohérente les préférences individuelles, maximisant ainsi la satisfaction globale de l'équipe.

3 Applications en optimisation pour l'e-commerce

3.1 Cas particulier 1.1

Le problème consiste à modéliser et résoudre l'affectation de demandes en fluide provenant de différentes commandes, en tenant compte des coûts unitaires associés à chaque magasin d'origine. Chaque magasin a une quantité limitée de stock disponible. L'objectif est de développer un programme linéaire pour optimiser la gestion de la livraison de fluides ou de colis, en fonction des contraintes spécifiques à chaque scénario.

Choix des Variables de Décision et Leur Domaine de Définition

Dans le modèle d'optimisation présenté, les variables de décision ont été judicieusement choisies pour représenter les quantités de fluide à affecter de chaque magasin (i) à chaque demande (k) pour chaque type de fluide (j). La variable `Quantites[i,j,k]` indique la quantité du fluide j provenant du magasin i pour la demande k.

Le domaine de définition des variables de décision est défini comme étant les nombres réels positifs (≥ 0), reflétant la nature des quantités de fluides qui ne peuvent être négatives. Ainsi, les variables `Quantites[i,j,k]` représentent les quantités de fluides affectées, et leur domaine de définition ≥ 0 garantit des solutions physiquement réalisables.

Cette représentation permet de modéliser de manière exhaustive les flux de fluides entre les magasins, les demandes et les types de fluides, tout en respectant les contraintes de stockage des magasins et les demandes spécifiques de chaque type de fluide.

Résultats obtenus :

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

FIGURE 4 – Les résultats obtenus

Résumé des Résultats :

Les quantités optimales de fluides affectées pour chaque magasin, type de fluide et demande spécifique sont obtenues après résolution du modèle d'optimisation. Voici un aperçu des résultats :

- Pour le magasin `M1` :

  Quantité de fluide `F1` pour la demande `D1` : `2`.
  Quantité de fluide `F1` pour la demande `D2` : `0.5`.
  Quantité de fluide `F2` pour la demande `D1` : `0`
  Quantité de fluide `F2` pour la demande `D2` : `1`.

- Pour le magasin `M2` :

  Quantité de fluide `F1` pour la demande `D1` : `0` .
  Quantité de fluide `F1` pour la demande `D2` : `0.5`.

Quantité de fluide F2 pour la demande D1 : 0 .
Quantité de fluide F2 pour la demande D2 : 1.

- Pour le magasin M3 :

  Quantité de fluide F1 pour la demande D1 : 0 .
  Quantité de fluide F1 pour la demande D2 : 0 .
  Quantité de fluide F2 pour la demande D1 : 0 .
  Quantité de fluide F2 pour la demande D2 : 1.

Analyse des résultats
Les résultats obtenus semblent cohérents et alignés avec les attentes du problème d'optimisation. En Effet, pour la demande D1 nécessitant 2 unités de F1, le magasin M1, ayant le coût unitaire le plus bas pour F1, attribue 2 unités de F1 à cette demande. De même, la demande D2 nécessitant 1 unité de F1 est attribuée en partie par M1 et le reste est pris en charge par M2 qui a le coût le plus bas pour F1 après M1.

La même logique s'applique à F2, où M1 satisfait autant que possible la demande D2,puis M3 satisfait autant que possible la demande D2 et le reste est pris en charge par M2.

Ainsi, la répartition des fluides entre les magasins semble optimale, minimisant le coût total tout en respectant les contraintes de stock disponibles pour chaque magasin. Il est important de noter que le coût total minimal obtenu est de 9.5, démontrant l'efficacité de la solution dans la gestion des ressources et la réduction des coûts.

3.2 Cas particulier 1.2

Le problème d'affectation consiste à optimiser la gestion de la livraison de fluides ou de colis, en tenant compte des coûts unitaires associés à chaque magasin d'origine. Chaque magasin dispose d'une quantité limitée de stock disponible, et l'objectif est d'attribuer de manière optimale les demandes en fluide provenant de différentes commandes tout en minimisant les coûts globaux. Cette optimisation inclut également des frais de livraison, à la fois fixes et variables.
Choix des Variables de Décision et Leur Domaine de Définition Dans le modèle d'optimisation présenté, les variables de décision ont été judicieusement choisies pour représenter les quantités de fluide à affecter de chaque magasin (i) à chaque demande (k) pour chaque type de fluide (j). La variable `Quantites[i,j,k]` indique la quantité du fluide j provenant du magasin i pour la demande k. **Contraintes :**

- Contrainte pour FraisExpedition nulle s'il n'y a pas de demande dans un magasin m :

$$\text{Contrainte 1 :} \quad FraisExpedition_{ij} = 0 \quad \text{si} \sum_k Quantites_{ijk} = 0 \quad \text{pour chaque } i, j \tag{1}$$

- Contrainte pour FraisExpedition non nulle s'il y a une demande dans un magasin m :

$$\text{Contrainte 2 :} \quad FraisExpedition_{ij} \geq F_{ij}^{fixe} + F_{ij}^{variable} \cdot \sum_k Quantites_{ijk} \quad \text{pour chaque } i, j \tag{2}$$

- Cette contrainte garantit que la somme des quantités expédiées depuis tous les magasins vers un client donné ne dépasse pas la capacité d'expédition de ce client, déterminée par

$$\text{Contrainte 3 :} \quad \sum_i Quantites_{ij} \leq \sum_i stock_{ij} \cdot FraisExpedition_j \quad \text{pour chaque } j \tag{3}$$

Résultats obtenus :

No.	Column name		Activity	Lower bound	Upper bound
1	Q[P1,M1,D1]	*	0	0	
2	Q[P1,M1,D2]	*	1	0	
3	Q[P1,M2,D1]	*	0	0	
4	Q[P1,M2,D2]	*	0	0	
5	Q[P1,M3,D1]	*	2	0	
6	Q[P1,M3,D2]	*	0	0	
7	Q[P2,M1,D1]	*	0	0	
8	Q[P2,M1,D2]	*	1	0	
9	Q[P2,M2,D1]	*	0	0	
10	Q[P2,M2,D2]	*	2	0	
11	Q[P2,M3,D1]	*	0	0	
12	Q[P2,M3,D2]	*	0	0	
13	FraisExpedition[D1,M1]	*	0	0	1
14	FraisExpedition[D2,M1]	*	1	0	1
15	FraisExpedition[D1,M2]	*	0	0	1
16	FraisExpedition[D2,M2]	*	1	0	1
17	FraisExpedition[D1,M3]	*	1	0	1
18	FraisExpedition[D2,M3]	*	0	0	1

FIGURE 5 – Les résultats obtenus

```
Problem:    PbEcomm1
Rows:       23
Columns:    18 (6 integer, 6 binary)
Non-zeros:  78
Status:     INTEGER OPTIMAL
Objective:  Cout = 368 (MINimum)
```

FIGURE 6 – Les résultats obtenus

3.3 Cas particulier 2

Le problème concerne la planification des livraisons pour un magasin ALPHA. L'objectif est de minimiser les coûts de livraison en déterminant l'itinéraire optimal pour un livreur. Formulé en programmation linéaire en nombres entiers (PLNE), le modèle prend en compte les relations entre le magasin, le livreur et les clients, assurant une visite unique à chaque client. La solution optimale, basée sur une matrice des distances, détermine l'itinéraire optimal du livreur.

Choix des Variables :

Nous avons choisi la variable binaire x_{ij} pour représenter si l'arc reliant les nœuds i et j est utilisé dans le parcours (1) ou non (0). Cette variable est cruciale pour déterminer le chemin emprunté par le livreur. De plus, nous avons introduit la variable entière u_i qui indique le rang du nœud i dans le parcours. Ces variables sont essentielles pour modéliser et résoudre le problème d'optimisation visant à minimiser la distance totale parcourue.

Contraintes :

- Sortir Une Seule Fois de Chaque Noeud :

$$\sum_{j \in \text{Noeuds}, j \neq i} x_{ij} = 1$$

- Visiter Une Seule Fois Chaque Nœud :

$$\sum_{i \in \text{Noeuds}, i \neq j} x_{ij} = 1$$

- Éviter Les Cycles :

$$u_i - u_j + \text{Card(Noeuds)} \cdot x_{ij} \leq \text{Card(Noeuds)} - 1$$

Résultats obtenus :

No.	Column name		Activity	Lower bound	Upper bound
1	x[ALPHA,C1]	*	1	0	1
2	x[ALPHA,C2]	*	0	0	1
3	x[ALPHA,C3]	*	0	0	1
4	x[ALPHA,C4]	*	0	0	1
5	x[ALPHA,C5]	*	0	0	1
6	x[C1,ALPHA]	*	0	0	1
7	x[C1,C2]	*	0	0	1
8	x[C1,C3]	*	1	0	1
9	x[C1,C4]	*	0	0	1
10	x[C1,C5]	*	0	0	1
11	x[C2,ALPHA]	*	1	0	1
12	x[C2,C1]	*	0	0	1
13	x[C2,C3]	*	0	0	1
14	x[C2,C4]	*	0	0	1
15	x[C2,C5]	*	0	0	1
16	x[C3,ALPHA]	*	0	0	1
17	x[C3,C1]	*	0	0	1
18	x[C3,C2]	*	0	0	1
19	x[C3,C4]	*	1	0	1
20	x[C3,C5]	*	0	0	1
21	x[C4,ALPHA]	*	0	0	1
22	x[C4,C1]	*	0	0	1
23	x[C4,C2]	*	0	0	1
24	x[C4,C3]	*	0	0	1
25	x[C4,C5]	*	1	0	1
26	x[C5,ALPHA]	*	0	0	1
27	x[C5,C1]	*	0	0	1
28	x[C5,C2]	*	1	0	1
29	x[C5,C3]	*	0	0	1
30	x[C5,C4]	*	0	0	1
31	u[C2]	*	5	1	6
32	u[C1]	*	1	1	6
33	u[C3]	*	2	1	6
34	u[C4]	*	3	1	6
35	u[C5]	*	4	1	6

FIGURE 7 – Les résultats obtenus

À partir des résultats obtenus, on observe que le chemin optimal pour le livreur, assurant une distance minimale, est le suivant : Alpha → C1 → C3 → C4 → C5 → C2 → Alpha. Ce chemin semble cohérent, car le livreur ne passe pas par un client plus d'une fois, il n'y a pas de cycles, et à la fin de sa livraison, il retourne directement chez Alpha. Il est également pertinent de noter que la distance minimale calculée pour ce chemin est de 22.