Université de Lille

Recherche Opérationnelle
Projet 1

Sous la direction de ***Bernard BECKERMANN***

Présenté par :
Carine BREIDY, Victor VANNOBEL

Année 2023-2024

M2 Ingénierie Statistique et numérique- Data Sciences

Table des matières

1 Introduction

L'objectif dans la recherche opérationnelle consiste à développer des méthodes efficaces pour résoudre des problèmes de maximisation ou de minimisation. Il y a par exemple des problèmes de cheminement dont le but est de chercher l'itinéraire de longueur minimale pour aller d'un point à un autre ; des problèmes de transport ou encore des problèmes de distribution de ressources. C'est d'ailleurs un problème de ce dernier type que nous allons traiter dans ce rapport.

AMPL est un logiciel qui permet de formuler mathématiquement des problèmes d'optimisation et de les résoudre. C'est ce logiciel que nous utiliserons au cours de ce projet.

Nous commencerons par expliquer plus en détails notre problème de minimisation de coûts total, puis nous discuterons de la modélisation mathématique : les variables, les paramètres, la fonction objectif ainsi que les contraintes. Nous pourrons ensuite exposer nos résultats qui vont définir de façon optimale la feuille de route à suivre.

2 Partie 1 : Etude du problème

2.1 Présentation du problème

Grâce à une modélisation sur AMPL, nous allons résoudre un problème de minimisation de coût, en prenant en considération la distribution d'un produit fabriqué dans deux usines vers différents clients, avec la possibilité d'utiliser ou non des dépôts intermédiaires. L'entreprise dispose de deux sites de production et de quatre dépôts pour servir six clients distincts. L'objectif de ce rapport est de proposer des solutions permettant de minimiser les coûts de transport.

Nous disposons de diverses données et contraintes pour organiser la livraison de nos produits. Cela implique de respecter les capacités de production de nos usines, de ne pas dépasser le débit mensuel maximal de nos dépôts, et surtout de répondre aux besoins de tous nos clients.

Chaque livraison a un coût, que ce soit du transport des usines vers les dépôts, des usines vers les clients, ou encore des dépôts vers les clients. Il est important de noter que toutes les combinaisons de routes ne sont pas possibles. Toutes les informations numériques nécessaires sont présentées dans les schémas ci-dessous.

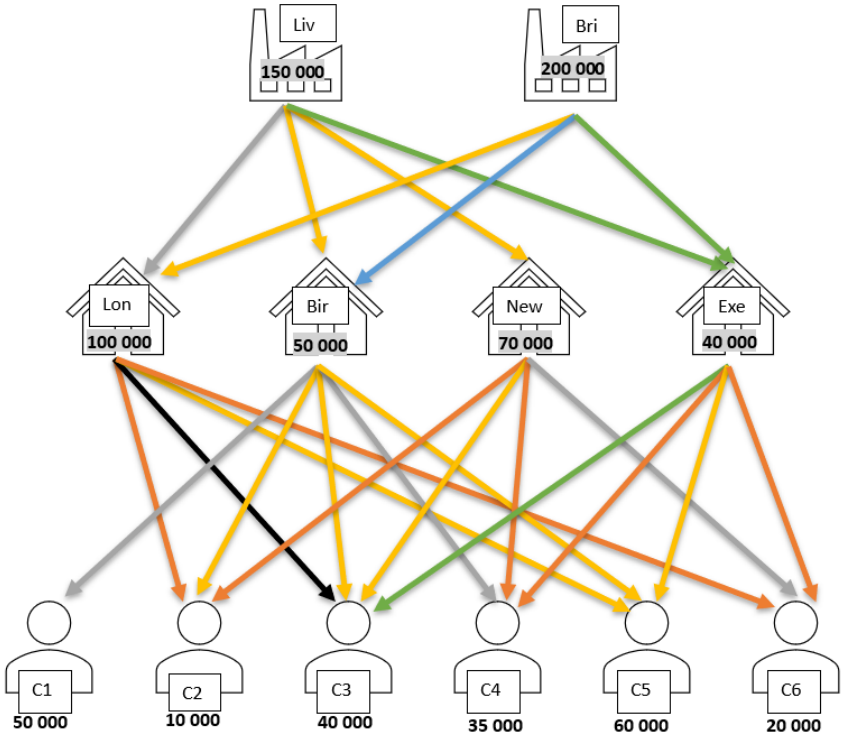

FIGURE 1 – Schéma récapitulatif 1

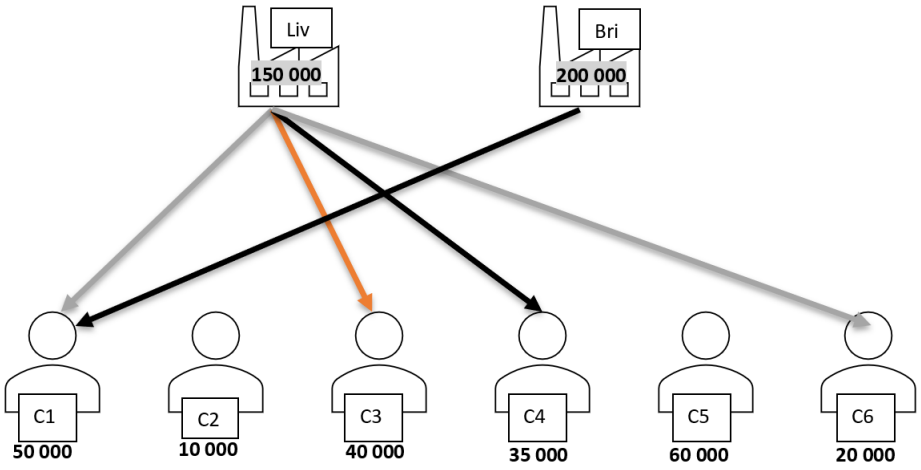

FIGURE 2 – Schéma récapitulatif 2

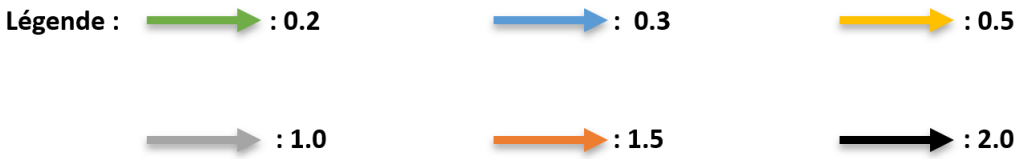

FIGURE 3 – Légende des deux schéma

La figure 1 montre les possibilités de livraison (usine,dépôt) et (dépôt, clients) et la figure 2 montre les possibilités de livraison (usine,clients). Chaque route est illustrée par une flèche colorée en fonction du coût de distribution légendé dans la figure 3.

Avant d'entrer dans la modélisation et sa résolution, il est possible de tirer des enseignements des schémas qui nous montrent des routes potentiellement plus efficaces, susceptibles de figurer dans la solution optimale. La solution la plus évidente serait de livrer le client 3 depuis le dépôt d'Exeter, après avoir récupéré les produits de l'une des deux usines. De plus, il est clair que si l'usine de Liverpool peut fournir directement les clients à un coût raisonnable, cela constituerait un avantage significatif.

Après cette brève présentation et ces premières observations, nous pouvons passer à la section suivante de ce rapport, où nous allons débuter en exposant le modèle AMPL de ce problème.

2.2 Problème en language AMPL

2.2.1 Les ensembles

Nous créons des ensembles qui permettent de ranger les usines, les dépôts, les clients et les routes souhaitées (de dimension 2).
set USINES :
Cet ensemble regroupe les lieux des usines : Liverpool "Liv" et Brighton "Bri".
set DEPOTS :
Cet ensemble regroupe les lieux des dépôtes : Newcastle "New", Birmingham "Bir",Londres "Lon" et Exeter "Exe".
set CLIENTS :
Cet ensemble regroupe les clients "C1", "C2","C3","C4","C5" et "C6".
set ROUTES :
Cet ensemble regroupe les routes (de dimension 2) entre (usines, dépôts), (dépôts,clients) et (usines,clients) : "(Liv,New)", "(Liv,Bir)", "(Liv,Lon)", "(Liv,Exe)", "(Liv,C1)", "(Liv,C3)", "(Liv,C4)", "(Liv,C6)", "(Bri,Bir)", "(Bri,Lon)", "(Bri,Exe)", "(Bri,C1)", "(New,C2)", "(New,C3)", "(New,C4)", "(New,C6)", "(Bir,C1)", "(Bir,C2)", "(Bir,C3)", "(Bir,C4)", "(Bir,C5)", "(Lon,C2)", "(Lon,C3)", "(Lon,C5)", "(Lon,C6)", "(Exe,C3)", "(Exe,C4)", "(Exe,C5)", "(Exe,C6)" .

2.2.2 Les paramètres

Concernant les paramètres, nous créons quatre paramètres qui sont tous positifs cités ci-dessous :
param couts {ROUTES} :
qui représente les coûts des différents routes possibles.
param cap_usi {USINES} :
qui représente la quantité maximale pouvant être envoyée par chaque usine.
param cap_dep{DEPOTS} :
qui représente la quantité maximale pouvant être reçue pour chaque dépôt.
param besoin {CLIENTS} :
qui représente les besoins des clients.

2.2.3 Les variables

Pour les variables, nous créons 29 variables positives (il y'a 29 routes possibles) que l'on nomme "tonnes{ROUTES}". Cela représente les tonnes à envoyer depuis les usines ou les dépôts.

2.2.4 La fonction à minimiser

L'objectif de cette modélisation vise à réduire au maximum les dépenses liées au transport des produits vers les clients. Pour ce faire, nous formulons une fonction objectif qui consiste à calculer la somme totale de tous les coûts de transport engagés lors des livraisons de produits. En langage AMPL, cette fonction objectif est exprimée comme suit :

```
minimize cout_total :
sum{(i,j) in ROUTES} couts [i,j] * tonnes [i,j]
```

2.2.5 Les contraintes

Il y'a 4 contraintes à appliquer aux variables afin de résoudre le problème :

```
subject to contrainte_cap_usines {i in USINES} :
sum{(i,j) in ROUTES} tonnes [i,j] <= cap_usi[i];
```
Il s'agit de la contrainte de production maximale des usines.

```
subject to contrainte_cap_depots {i in DEPOTS} :
sum{(j,i) in ROUTES :j in USINES } tonnes [j,i] <= cap_dep[i];
```
Il s'agit de la contrainte de débit maximal des dépots.

```
subject to contrainte_besoins {j in CLIENTS} :
sum{(j,i) in ROUTES} tonnes [i,j] <= besoin[j];
```
Il s'agit de la contrainte de satsisfaction des demandes des clients.

```
subject to contrainte_stock_depots {i in DEPOTS} :
sum{(j,i) in ROUTES :j in USINES } tonnes [j,i] >= sum{(i,k) in ROUTES :k in DEPOTS } tonnes
[i,k];
```
Il s'agit de la contrainte pour modéliser le fait que les dépôts ne peuvent envoyer plus que ce qu'ils reçoivent des usines.

Au fur et à mesure de l'avancement du projet, diverses contraintes et variables seront ajoutées afin de résoudre les différents problèmes.

2.3 Résultats

Après résolution du problème à l'aide du logiciel AMPL, on trouve un coût total minimum de 198 500 livres. Le schéma de distribution est le suivant : (les valeurs sont données en milliers de tonnes)

```
tonnes [*,*] (tr)
:    Bir  Bri  Exe  Liv  Lon New
Bir   .    50   .    0    .    .
C1    0    0    .    50   .    .
C2    10   .    .    .    0    0
C3    0    .    40   0    0    0
C4    35   .    0    0    .    0
C5    5    .    0    .    55   .
C6    .    .    0    20   0    0
Exe   .    0    .    40   .    .
Lon   .    55   .    0    .    .
New   .    .    .    0    .    .
```

FIGURE 4 – Schéma de distribution optimal.

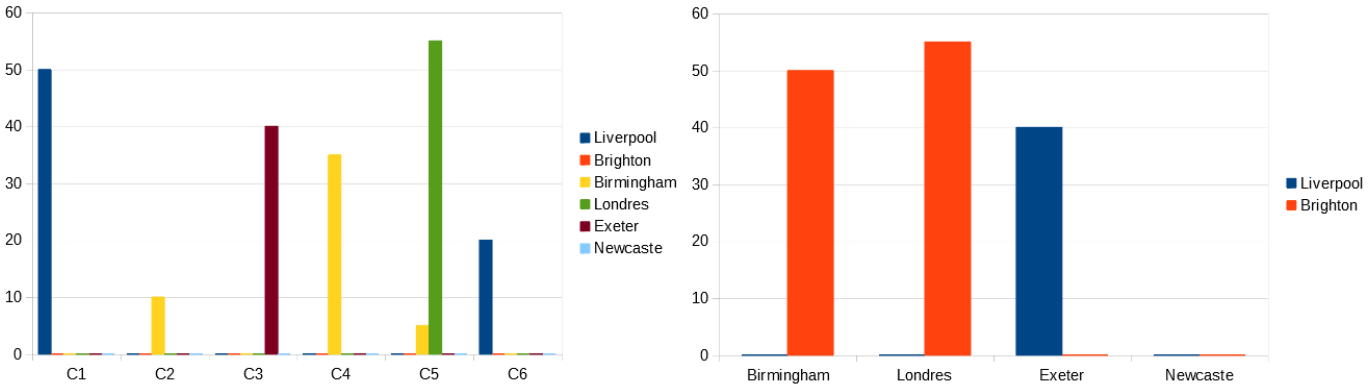

FIGURE 5 – Graphiques des tonnes envoyées aux dépôts et aux clients.

Les figures 4 et 5 montrent que dans le cas du schéma de distribution optimal, les usines n'atteignent pas leurs capacités maximales de production. On peut quand même noter que l'usine de Liverpool utilise 73.33% de sa capacité de production totale alors que l'usine de Brighton n'en utilise que 52.5%.
Au contraire, les dépôts de Exeter et Birminghamn ont atteint leurs limites de tonnes provenant des usines.
Le dépôt de Newcastle n'est pas utilisé, ce n'est pas rentable.

On peut donc se demander ce qu'il se passerait si les capacités des usines et des dépôts augmentaient (on s'attend à avoir une réduction du coût total).

3 Modification des données

3.1 Variations des données

Nous allons discuter dans cette partie de l'impact d'un changement des capacités de production et de réceptions des usines et dépôts sur le schéma de distribution ainsi que sur le coût total.

On remarque que les usines n'ont pas atteint leurs capacités maximales de productions et que les dépôts de Exeter et Birminghamn ont atteint leurs capacités maximales de réception de tonnes. En supposant qu'accroître les capacités des usines et dépôts ait un prix, il serait judicieux de ne pas accroître directement les capacités de toutes les usines ni celles de tous les dépôts exceptés ceux de Exeter et de Birmingham.

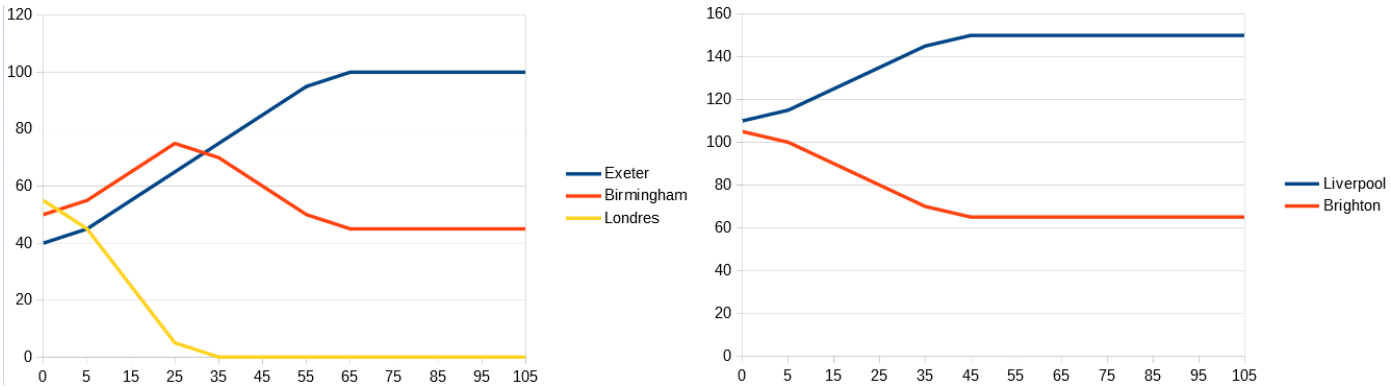

FIGURE 6 – Evolution de l'usage des dépôts en fonction de l'augmentation de la capacité.

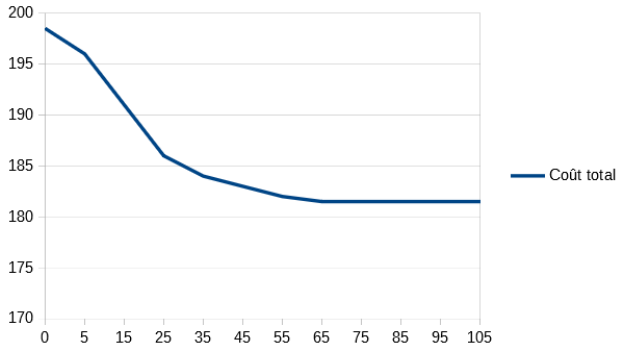

FIGURE 7 – Evolution du coût total en fonction de l'augmentation de la capacité de certains dépôts.

On s'aperçoit lorsque la capacité des dépôts Exeter et Birmingham augmente que l'on préfère fournir des tonnes avec l'usine de Liverpool plutôt qu'avec celle de Birmingham. Il serait donc judicieux d'augmenter la capacité de production de l'usine de Liverpool.
On s'apeçoit également que le dépot d'Exeter est le plus rentable et que le dépôt de Londres ne reçoit plus de marchandises dès lors que les 2 dépôts d'Exeter et Birmingham ont été accrus de 35

000 tonnes chacuns, tout comme le dépôt de Newcastle depuis le départ.
Le coût total diminue au fur et à mesure que les capacités augmentent (figure 7).

Nous allons maintenant augmenter la capacité de production de l'usine de Liverpool en plus.

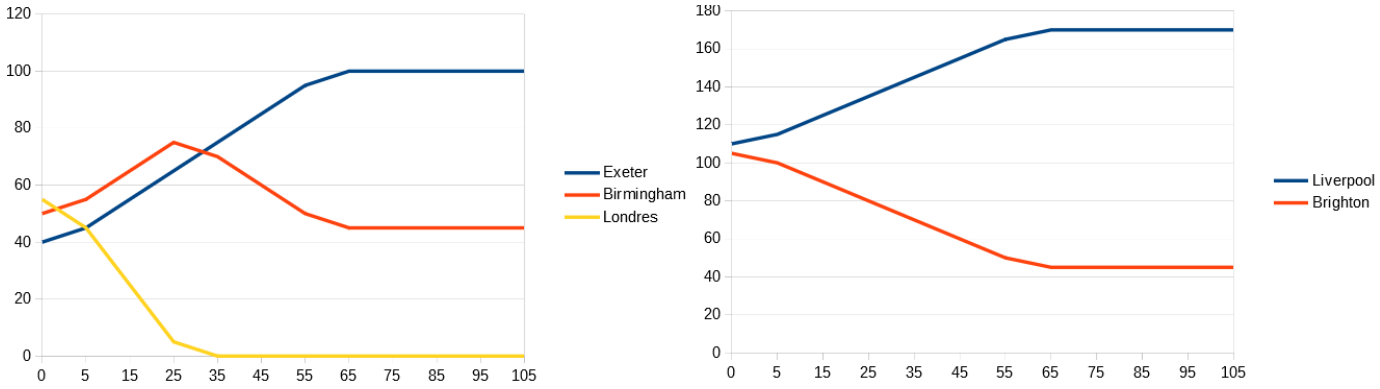

FIGURE 8 – Evolution de l'usage de certains dépôts et de l'usine de Liverpool en fonction de l'augmentation de la capacité.

D'après la figure 8, l'évolution du coût total est la même que celle de la figure 6. On obersve que l'évolution des marchandises reçues par les dépôts est aussi la même que dans le cas précédent malgré l'augmentation de la production maximale possible de l'usine de Liverpool. Cela s'explique par le fait que le coût de transport de marchandises pour envoyer de Brighton ou Liverpool vers Exeter est le même.
Il n'est donc pas nécessaire d'augmenter la capacité de production des usines pour réduire les dépenses. De plus, on constate toujours que plus la capacité du dépôt d'Exeter est importante, moins le dépôt de Birmingham est solicité.

On décide finalement d'augmenter uniquement la capacité de réception du dépôt d'Exeter de 60 000 tonnes soit au total 100 000 tonnes. On obtient le schéma de distribution suivant (figure 9) :

```
tonnes [*,*] (tr)
:    Bir  Bri  Exe  Liv Lon New
Bir   .    45   .     0   .   .
C1    0    0    .    50   .   .
C2   10    .    .     .   0   0
C3    0    .   40     0   0   0
C4   35    .    0     0   .   0
C5    0    .   60     .   0   .
C6    .    .    0    20   0   0
Exe   .   20    .    80   .   .
Lon   .    0    .     0   .   .
New   .    .    .     0   .   .
```

FIGURE 9 – Schéma de distribution optimal après augmentation de la capacité du dépôt d'Exeter.

Le coût total est de 181 500 livres soit 17 000 livres de moins qu'avant augmentation de la capacité de stockage du dépôt d'Exeter. Les dépôts de Londres et de Newcastle ne reçoivent ni n'envoient de marchandises. Il peut donc être intéressant de fermer ces 2 dépôts et de payer pour aggrandir la capacité maximale du dépot d'Exeter.

3.2 Satisfaction des clients

Plus de la moitié des clients souhaitent être livrés par des fournisseurs en particulier. On souhaite ici savoir s'il est possible de satisfaire les préférences de tous les clients.

Après observations des préférences des clients et des débits maximums des dépôts, on observe facilement qu'il n'est pas possible de satisfaire les préférences des clients puisque le client 5 exige 60 000 tonnes provenant du dépôt de Birminghan alors que ce dernier ne peut recevoir que seulement 50 000 tonnes des usines.

On décide donc d'ignorer les préférences du client 5 et il s'avère que satisfaire les préférences des 5 autres clients est réalisable. On obtient le schéma de distribution suivant :

```
tonnes [*,*] (tr)
:    Bir  Bri  Exe  Liv  Lon  New
Bir   .    50   .    0    .    .
C1    0    0    .    50   .    .
C2    0    .    .    .    0    10
C3    0    .    40   0    0    0
C4    35   .    0    0    .    0
C5    15   .    0    .    45   .
C6    .    .    0    0    20   0
Exe   .    40   .    0    .    .
Lon   .    65   .    0    .    .
New   .    .    .    10   .    .
```

FIGURE 10 – Schéma de distribution optimal pour satisfaire les préférences des clients (sauf le 5-ème).

D'après la figure 10, les exigences des clients (excepté le 5-ème) quant aux fournisseurs sont réalisables pour un coût total minimum de 228 500 livres soit 30 000 livres de plus qu'en les ignorant. On constate que tous les dépôts sont solicités.

On peut maintenant se demander ce qu'il se passerait si on complétait partiellement les préférences du client 5, c'est à dire lui fournir 50 000 tonnes du dépot de Birmingham puis de compléter avec des tonnes d'autres sources.

```
tonnes [*,*] (tr)
:    Bir  Bri  Exe  Liv  Lon  New
Bir   .    50   .    0   .    .
C1    0    0   .    50  .    .
C2    0    .   .    .    0   10
C3    0    .   40   0    0    0
C4    0    .    0   35  .    0
C5   50    .    0   .   10   .
C6    .    .    0    0  20    0
Exe   .    40   .    0   .    .
Lon   .    30   .    0   .    .
New   .    .    .   10   .    .
```

FIGURE 11 – Schéma de distribution optimal pour satisfaire au mieux les préférences des clients.

On observe d'après la figure 11 que les préférences du client 5 ne sont pas totalement satisfaites, il aura fallu envoyer 10 000 tonnes depuis le dépôt de Londres en plus de celles du dépôt de Birmingham pour minimiser le coût total.

Le coût total minimum revient à 246 000 livres soit 17 500 livres de plus qu'en ignorant totalement les préférences de C5 ou encore 47 500 livres de plus que le cas original.

3.3 Modification des dépôts

On souhaite ici étudier la possibilité pour l'entreprise d'effectuer différentes modifications au sein des dépôts. En effet, il est possible d'ouvrir 2 nouveaux dépôts à Bristol et à Northampton, d'agrandir celui de Birmingham ou enfin de fermer ceux de Newcaste et d'Exeter. L'entreprise considère qu'il n'est pas souhaitable de maintenir plus de 4 dépôts.

En introduisant des variables binaires, nous aurons la possibilité de voir quels changements permettent d'avoir le coût minimum :

```
statut_dep [*]
  Bir  1
Brist  0
  Exe  0
  Lon  0
  New  1
North  1
```

FIGURE 12 – Variables indiquant s'il y'a eu un changement.

Après résolution, on trouve un coût total minimum de 174 000 livres soit 24 500 livres de moins que dans le cas de départ. On observe que le dépôt de Birmingham a été agrandi, que le dépôt de Northampton a été crée et que le dépôt de Newcastle a été fermé, ce qui semble logique car il n'était pas utilisé dans le cas initial et l'entreprise gagne une certaine somme à le fermer...

```
tonnes [*,*] (tr)
:      Bir  Bri Brist  Exe  Liv  Lon New North
Bir      .    70     .    .    0    .    .     .
Brist    .     0     .    .    0    .    .     .
C1       0     0     0    .   50    .    .     .
C2      10     .     0    .    .    0    0     0
C3       0     .     0   40    0    0    0     .
C4      10     .     .    0    0    .    0    25
C5      50     .     0    0    .   10    .     0
C6       .     .     0    0   20    0    0     0
Exe      .    40     .    .    0    .    .     .
Lon      .    10     .    .    0    .    .     .
New      .     .     .    .    0    .    .     .
North    .    25     .    .    0    .    .     .
```

FIGURE 13 – Schéma de distribution optimal après modification des dépôts.

On constate que le dépôt de Birmingham est pleinement solicité malgré l'extension. C'est normal puisqu'il s'agit globalement du dépôt le plus rentable derrière celui d'Exeter (il est évident d'après la précédente étude qu'il n'est pas rentable de fermer ce dernier).

4 Conclusion

Ce projet a permis d'utiliser la méthode du Simplex via le logiciel AMPL pour traiter un problème d'optimisation de transport de marchandises. Le logiciel AMPL a été utile pour obtenir facilement un large panel de données servant à la mise en oeuvre de tableaux et de graphes.
L'analyse en détail du problème a permis de montrer que certaines modifications judicieuses des paramètres pouvait réduire les dépenses, notamment les capacités de débits maximums des dépots où il a été facile de s'en rendre compte grâce à la possibilité que nous donne le logiciel AMPL de faire varier facilement les données.

Il a également été intéressant d'introduire des variables bivalentes décisionnelles afin de faire la bonne décision quant à des modifications sur les dépôts pour réduire les dépenses.

5 Annexes

Fichier .mod

```
set USINES; #ensemble des usines
set DEPOTS; #ensemble des depots
set CLIENTS; #ensemble des clients

set ROUTES dimen 2; #ensemble des routes possibles, 2 dimensions
set ROUTES_NON_SOUHAITEES dimen 2; #q3 pour satisfaire les clients
set ROUTES_q3 := ROUTES diff ROUTES_NON_SOUHAITEES; #q3

param couts{ROUTES} >= 0; #les couts des differentes routes possibles
param cap_usi{USINES} >= 0; #quantite maximale pouvant etre envoyee par chaque usine
param cap_dep{DEPOTS} >= 0; #quantite maximale pouvant etre recue pour chaque depot
param besoin{CLIENTS} >= 0; #besoins des clients

#question 4
param cap_depart{DEPOTS} >= 0; #capacite initiale des depots avant tout changement
param cap_chgmt{DEPOTS}; #capacite apres changement
param cout_chgmt{DEPOTS}; #cout du changement

var tonnes{ROUTES} >= 0; #les tonnes a envoyer depuis les usines/depots
#var statut_dep{DEPOTS} binary; #q4, 0 si pas de chgmt, 1 sinon

#pour les questions 1/2
minimize cout_total : #fonction a minimiser
        sum{(i,j) in ROUTES} couts[i,j] * tonnes[i,j];

subject to contrainte_cap_usines {i in USINES}:
        sum{(i,j) in ROUTES} tonnes[i,j] <= cap_usi[i];
#quantite maximale pouvant etre envoyee depuis les usines

subject to contrainte_cap_depots {i in DEPOTS}: #quantite maximale pouvant etre recue
        sum{(j,i) in ROUTES : j in USINES} tonnes[j,i] <= cap_dep[i];

subject to contrainte_besoins {j in CLIENTS}: #quantite demandee par les clients
         sum{(i,j) in ROUTES} tonnes[i,j] = besoin[j];

subject to contrainte_stock_depots {i in DEPOTS}:
sum{(j,i) in ROUTES : j in USINES} tonnes[j,i]
>= sum{(i,k) in ROUTES : k in CLIENTS} tonnes[i,k];
#les depots ne peuvent pas envoyer plus que ce qu'ils recoivent depuis les usines



#pour la question 3
#minimize cout_total : #fonction a minimiser
#       sum{(i,j) in ROUTES_q3} couts[i,j] * tonnes[i,j];

#subject to contrainte_cap_usines {i in USINES}:
#       sum{(i,j) in ROUTES_q3} tonnes[i,j] <= cap_usi[i];
#quantite maximale pouvant etre envoyee depuis les usines, q3
```

```
#subject to contrainte_cap_depots {i in DEPOTS}:
#       sum{(j,i) in ROUTES_q3 : j in USINES} tonnes[j,i] <= cap_dep[i];
#quantite maximale pouvant etre recue par les depots depuis les usines

#subject to contrainte_besoins {j in CLIENTS}: #quantite demandee par les clients, q3
#        sum{(i,j) in ROUTES_q3} tonnes[i,j] = besoin[j];

#subject to contrainte_stock_depots {i in DEPOTS}:
#sum{(j,i) in ROUTES_q3:j in USINES} tonnes[j,i]
>=sum{(i,k) in ROUTES_q3:k in CLIENTS} tonnes[i,k];
#les depots ne peuvent pas envoyer plus que ce qu'ils recoivent depuis les usines

#subject to contrainte_client5 :
#       tonnes["Bir","C5"] = 50;
#satisfaire partiellement les preferences du client 5



#pour la question 4
#minimize cout_total :
#sum{(i,j) in ROUTES} couts[i,j]*tonnes[i,j]
#+sum{d in DEPOTS} cout_chgmt[d]*(statut_dep[d]);

#subject to contrainte_cap_usines {i in USINES}:
#       sum{(i,j) in ROUTES} tonnes[i,j] <= cap_usi[i];
#quantite maximale pouvant etre envoyee depuis les usines

#subject to contrainte_cap_depots {i in DEPOTS}:
#sum{(j,i) in ROUTES} tonnes[j,i] <= cap_depart[i] + (statut_dep[i])*cap_chgmt[i];

#subject to contrainte_besoins {j in CLIENTS}: #quantite demandee par les clients
#        sum{(i,j) in ROUTES} tonnes[i,j] = besoin[j];

#subject to contrainte_stock_depots {i in DEPOTS}:
#sum{(j,i) in ROUTES : j in USINES} tonnes[j,i]
>= sum{(i,k) in ROUTES : k in CLIENTS} tonnes[i,k];
#les depots ne peuvent pas envoyer plus que ce qu'ils recoivent depuis les usines

#subject to contrainte_depots_max : #pas plus de 4 depots
#       sum{i in DEPOTS} statut_dep[i] <= 4;
```

Fichier .dat

```
set USINES := Liv Bri ;
set DEPOTS := New Bir Lon Exe
                    #Brist North #q4
                    ;
set CLIENTS := C1 C2 C3 C4 C5 C6;

set ROUTES := (Liv,New) (Liv,Bir) (Liv,Lon) (Liv,Exe)
              (Liv,C1) (Liv,C3) (Liv,C4) (Liv,C6)
              (Bri,Bir) (Bri,Lon) (Bri,Exe) (Bri,C1)
              (New,C2) (New,C3) (New,C4) (New,C6)
```

```
              (Bir,C1) (Bir,C2) (Bir,C3) (Bir,C4) (Bir,C5)
              (Lon,C2) (Lon,C3) (Lon,C5) (Lon,C6)
              (Exe,C3) (Exe,C4) (Exe,C5) (Exe,C6)
             #(Liv,Brist) (Bri,Brist) (Liv,North) (Bri,North) #q4
              #(Brist, C1) (Brist, C2) (Brist, C3) (Brist, C5) (Brist, C6) #q4
              #(North,C2) (North,C4) (North,C5) (North,C6) #q4
             ;

set ROUTES_NON_SOUHAITEES := (Bri,C1) (Bir,C1) (Lon,C2) #question 3
              (Bir,C2) (Liv,C6) (New,C6) ;
#ajouter (Lon,C5) (Exe,C5) pour montrer que la resolution est impossible

param cap_usi := Liv 150 Bri 200;
param cap_dep :=  New 70 Bir 50 Lon 100 Exe 40;
param besoin := C1 50 C2 10 C3 40 C4 35 C5 60 C6 20;

param cap_depart := New 70 Bir 50 Lon 100 Exe 40 Brist 0 North 0; #q4
param cap_chgmt :=  New -70 Bir 20 Lon 0 Exe -40 Brist 30 North 25; #q4
param cout_chgmt :=  New -10 Bir 3 Lon 0 Exe -5 Brist 12 North 4; #q4

param   couts  :=
Liv  New  0.5
Liv  Bir  0.5
Liv  Lon  1.0
Liv  Exe  0.2
Liv  C1   1.0
Liv  C3   1.5
Liv  C4   2.0
Liv  C6   1.0
Bri  Bir  0.3
Bri  Lon  0.5
Bri  Exe  0.2
Bri  C1  2.0
New  C2  1.5
New  C3  0.5
New  C4  1.5
New  C6  1.0
Bir  C1  1.0
Bir  C2  0.5
Bir  C3  0.5
Bir  C4  1.0
Bir  C5  0.5
Lon  C2  1.5
Lon  C3  2.0
Lon  C5  0.5
Lon  C6  1.5
Exe  C3  0.2
Exe  C4  1.5
Exe  C5  0.5
Exe  C6  1.5
#Liv   Brist  0.6 #q4
#Bri   Brist  0.4 #q4
#Liv   North  0.4 #q4
#Bri   North  0.3 #q4
```

```
#Brist C1  1.2 #q4
#Brist C2  0.6 #q4
#Brist C3  0.5 #q4
#Brist C5  0.3 #q4
#Brist C6  0.8 #q4
#North C2  0.4 #q4
#North C4  0.5 #q4
#North C5  0.6 #q4
#North C6  0.9 #q4
;
```

Fichier .run

```
# automatically generated on Wed Sep 27 2023 at 18:35:03 PM CEST
reset;
model ../monampl/projet-m2/exo2-victorcarine/exo2.mod;
data ../monampl/projet-m2/exo2-victorcarine/exo2.dat;
option solver cplex;
solve;

display tonnes;
#display statut_dep; #q4
display cout_total;

#pour la question 2, tester differentes modifications...
#set valeurs :=5..55 by 10;
#param cap_u{USINES};
#param cap_d{DEPOTS};
#param objectif{valeurs};
#let {k in USINES} cap_u[k] := cap_usi[k];
#let {l in DEPOTS} cap_d[l] := cap_dep[l];
#param lambda;

#for{i in valeurs}{
#       let lambda := i;
        #if(lambda >= 45) then {let cap_usi["Liv"] := cap_u["Liv"]+20;}

        #for{d in DEPOTS}{
        #       let cap_dep[d] := cap_d[d]+lambda;
        #}
#       solve;
#       let objectif[i] := cout_total;
#       display lambda;
#       display tonnes;
#       display cap_usi;
#       display cap_dep;
#       display objectif[i];
#}
```