Rapport du Miniprojet REOP 2020 - 2021

Manal Belhamadia, Mofan Zhang, Yunlu Zhao - Groupe 17 (25)

14 janvier 2021

1 Introduction

Modélisation du problème de dispatch des emballages propres et vides chez Renault :
On cherche à optimiser le dispatch des différents types d'emballages utilisés par les fournisseurs de Renault pour envoyer des pièces vers les usines sur un horizon temporel fixé.

2 Réponses aux questions

2.1 Cas d'un seul type d'emballage

Sous l'hypothèse $E = 1$ et $l_1 > L/2$, nous pouvons modéliser le problème comme un problème de b-flot dont le graphe est représenté par la figure 1. Nous considérons dans la figure une durée de 3 jours pour simplifier la modélisation. Dans le graphe, le flot est une circulation à partir du sommet source s à des nœuds intermédiaires jusqu'au sommet sink t et puis tout revient à s. Ainsi, tous les nœuds ont un paramètre $b = 0$. Pour chaque usine, le nœud du jour 0 représente le stock initial, entre le nœud de l'usine du jour j et du jour $j + 1$, nous utilisons deux nœuds supplémentaires pour représenter le stock inférieur ou égal au stock maximum ainsi que le stock excédentaire qui génère un coût de stock excédentaire. La source s est reliée à des sommets d'usine du jour j et chaque arête (s, u_j) a une capacité fixée (la capacité inférieure égale à la capacité supérieure) égale au nombre d'emballages libérés par l'usine le jour j, sauf pour l'arête de s à l'usine du jour 0 qui représente le stock initial du type d'emballage considéré.

Nous modélisons de manière identique le stock d'un fournisseur. En ce qui concerne la consommation de fournisseur, nous relions ses nœuds avec le sommet fictif t, et nous fixons la capacité de ces arêtes à b^-_{efj} qui représente le nombre d'emballages à consommer par chaque fournisseur chaque jour. Chaque nœud de fournisseur a une arête entrante qui permet de modéliser le fait qu'un fournisseur ne consomme que les emballages déjà disponibles dans le stock, sans considérer ceux livrés le même jour. Au cas où le stock de fournisseur n'est pas suffisant, le fournisseur utilise les cartons dont le flot est modélisé par le sommet fictif "source de cartons" et les arêtes vertes dans le graphe ci-dessous. Le nombre de cartons utilisés n'est pas limité mais chaque unité a un coût c^{exc}_{ef}.

2.2 Nombre de variables de la formulation PLNE (11) fournie

Une route R est un triplet (j_R, P_R, q_R). Dans le graphe $(U + F, A)$, le nombre de chemins admissibles est de $|U| \times \sum_{i=1}^{4} A_i^{|F|}$ dont $A_i^{|F|}$ est le nombre de permutations de i parmi $|F|$. Cette valeur est évidemment polynomial en fonction de $|U|$ et de $|F|$.

Le vecteur de chargement q_R a $|E|$ éléments. Étant donné que $|E|$, $|J|$, L et l_e sont tous fixés, le nombre de routes $|J| \times |P_R| \times |q_R|$ est donc polynomial en la taille de l'instance. Le nombre de routes est égal au nombre de variables x_R.

Pour l'instance *europe.csv*, nous avons 25 usines et 576 fournisseurs. Par conséquent, le nombre de variables de ce PLNE est bien supérieur à la limite de 1 million. Il n'est pas vraisemblable de résoudre cet instance avec ce PLNE.

2.3 Formulation PLNE du problème avec l'hypothèse du nombre de camion limité

Nous introduisons une suite de variables (y_{Ri}) dont la définition est :

$$y_{Ri} = \begin{cases} 1 & \text{Si } R \text{ passe par l'arête } i \text{ du graphe } (U+F, A) \\ 0 & \text{Sinon} \end{cases} \tag{1}$$

Un chemin couvre au plus 4 fournisseurs. La contrainte sur cette variable est donc :

$$\sum_i y_{Ri} \leq 4 \tag{2}$$

Pour un graphe constitué de toutes les usines et tous les fournisseurs, le nombre d'arête est inférieur ou égal à $|U+F|^2$.

Nous utilisons une variable z_{kR} pour noter si un camion k suit la route R :

$$z_{kR} = \begin{cases} 1 & \text{Si le camion } k \text{ suit la route } R \\ 0 & \text{Sinon} \end{cases} \tag{3}$$

Nous avons une contrainte sur le nombre de camions utilisés au jour j :

$$\sum_{R \in R_j} \sum_{k \in K} z_{kR} \leq K \quad R_j \text{ est l'ensemble de routes effectuées au jour } j, \ \forall j \in J \tag{4}$$

L'objectif de ce PLNE est donc :

$$\begin{aligned} min \sum_R c_R \sum_{k \in K} z_{kR} + \sum_{euj} c^s_{eu} \max(s_{euj} - r_{euj}, 0) \\ + \sum_{efj} c^s_{ef} \max(s_{efj} - r_{efj}, 0) + \sum_{efj} c^{exc}_{ef} \max(b^-_{efj} - s_{ef(j-1)}, 0) \end{aligned} \tag{5}$$

où le coût de dispatching est :

$$c_R = c^{cam} + c^{stop} \sum_i y_{Ri} + \gamma \sum_i d_i \times y_{Ri} \tag{6}$$

Les autres contraintes restent identiques que le PLNE fourni dans l'énoncé. Le nombre de routes est de l'ordre de $O(|J| \times |E| \times |K|)$. Le nombre de variables y_{Ri} est donc de l'ordre $O(|R| \times |U+F|^2)$. Ainsi, ce PLNE comprend $O(|J| \times |U+F|^2 \times |E| \times |K|)$ variables.

3 Algorithme

Afin de résoudre l'instance *europe.csv*, nous décomposons le problème initial en 2 sous-problèmes qui peuvent être résolus l'un après l'autre :

1. Un sous-problème de flot pour déterminer la quantité quotidienne expédiée de chaque type d'emballage de chaque usine à chaque fournisseur ;
2. Un sous-problème de tournée pour trouver une organisation acceptable de camions et de routes qui permet de satisfaire tous les besoins d'expédition des usines.

Une telle décomposition rend le problème plus simple à résoudre et peut nous donner une solution approximative. Pour le premier sous-problème, nous pouvons le modéliser comme un problème de flot et puis résoudre avec un solveur, alors que pour le deuxième, nous choisissons d'utiliser une méthode constructive pour trouver une solution initiale et de l'améliorer avec une recherche locale.

3.1 Modélisation PLNE d'un problème de flot pour déterminer la quantité d'expédition des usines aux fournisseurs

Nous modélisons le premier sous-problème d'une manière différente que celle de la question 1. Pour chaque type d'emballage, nous considérons le graphe de flot montré dans la figure 2 où il n'y a pas de circulation. Dans ce graphe, nous prenons comme convention que les nœuds de source ont un paramètre $b < 0$, alors que les nœuds de sink ont un paramètre $b > 0$. Pour le nœud de source de cartons, le paramètre b est indéfini parce que nous ne connaissons pas le nombre de cartons utilisés. Pour le dernier jour (le jour 3 dans le graphe), les nœuds de stock des usines (resp. des fournisseurs) ont $b \geq 0$ qui représentent le fait que le stock à la fin ne peut pas être négatif.

Nous pouvons résoudre le PLNE associé à ce graphe avec un solveur (Gurobi dans notre cas). Pour l'instance *europe.csv*, nous découpons la durée totale (21 jours) en trois plus petites périodes (7 jours chacune) et utilisons cette modélisation pour les résoudre une par une. Nous obtenons au final une matrice de taille $[E, U, F, J]$ dont chaque élément représente le nombre d'emballage de type e expédié de l'usine u au fournisseur f le jour j.

3.2 Méthode constructive pour trouver une solution initiale

Après avoir le résultat du sous problème de flot, on sait quelle usine expédie combien d'emballage auxquels fournisseurs, le problème est donc d'organiser les camions pour satisfaire les demandes avec un coût minimisé. Afin de construire une solution initiale qui donne un bon résultat, nous avons considérer l'algorithme greedy pour chaque usine et chaque jour j. Nous avons considéré deux aspects principales :

1. **First-fit decreasing :** pour la recharge au fournisseur f, le FFD peut être un bon choix. Mais dans le contexte du projet, il faut ajouter la contrainte de 4 arrêts au maximum pour chaque camion. Ainsi, on peut mettre un emballage dans un camion seulement si ce camion n'a pas encore 4 arrêts et qu'il y a suffisamment d'espace pour cet emballage.
2. **la recherche des fournisseurs à passer prochainement :** afin de minimiser le coût des kilomètres parcourus, un camion va passer le fournisseur le plus proche. Ce processus se réalise par la fonction *sort*.
3. **Des fournisseurs qui ont une demande importante :** On a d'abord implanté FFD pour tous les fournisseurs, mais gardé finalement seulement les camions qui sont bien remplis, qui est décidé par la variable r dans le code.

Le pseudo code de la stratégie heuristique est en-dessous :

Algorithm 1: FFD avec la recherche greedy

Input: u, j, Matrice de flot
Output: camions

$camions \leftarrow \emptyset$, $camions\_space \leftarrow \emptyset$, $rest\_items \leftarrow \emptyset$;
`/* 1. Expédier les gros fournisseurs avec First-Fit Decreasing */`
Expédier tous les f dans F_list avec FFD, garder les camions remplis à r, return *camions* et *camions_space* ;
`/* 2. Expédier le reste avec First-Fit Decreasing en ajoutant le contraint de 4 stops au maximum */`
while F_list **do**
  $f \leftarrow$ le f plus proche de $Point\_depart$;
  Expédier pour f
  `/* Actualiser Point_depart et F_list */`
  $Point\_depart \leftarrow f$;
  Supprimer f de F_list
end

Grâce à cet algorithme, nous avons obtenu une solution de bon résultat pour chaque instance fournie :

Instance	Europe	France	Espagne	Maroc
Coût total	7 789 506	2 090 854	1 322 729	368 509

3.3 Recherche locale pour la planification de routes

Dans cette partie, on essaie de trouver des voisins de la solution initiale et de garder celui qui donne le coût minimal. Nous avons implanter deux façon pour trouver des voisins plus performants :

1. Supprimer des routes
2. Changer l'ordre de stops pour chaque route

Méthode appliquée	1 : Suppression de routes	2 : Changement de l'ordre de stops	1 + 2
Coût avant	7 789 506	7 789 506	7 599 696
Coût après	7 599 696	7 027 523	6 682 930

4 Conclusion

Nous avons fait le choix de séparer le problème en deux sous-problèmes. Avec la modélisation du premier PLNE en problème de flot et avec l'utilisation des méthodes heuristiques pour la recherche d'une solution acceptable pour la constitution des routes, on obtient une borne supérieure des solutions possibles. Il aurait sûrement été plus efficace de les faire interagir afin de mieux affiner la constitution des routes pour la livraison des emballages.

5 Annexe

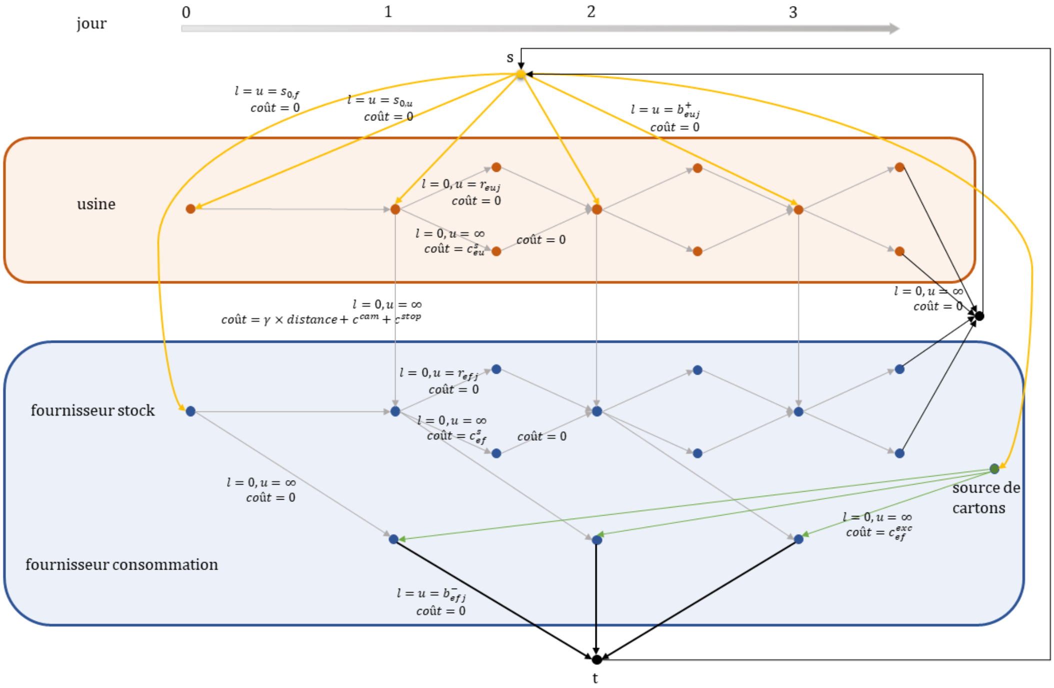

Figure 1 – Graphe b-flot pour une durée de 3 jours

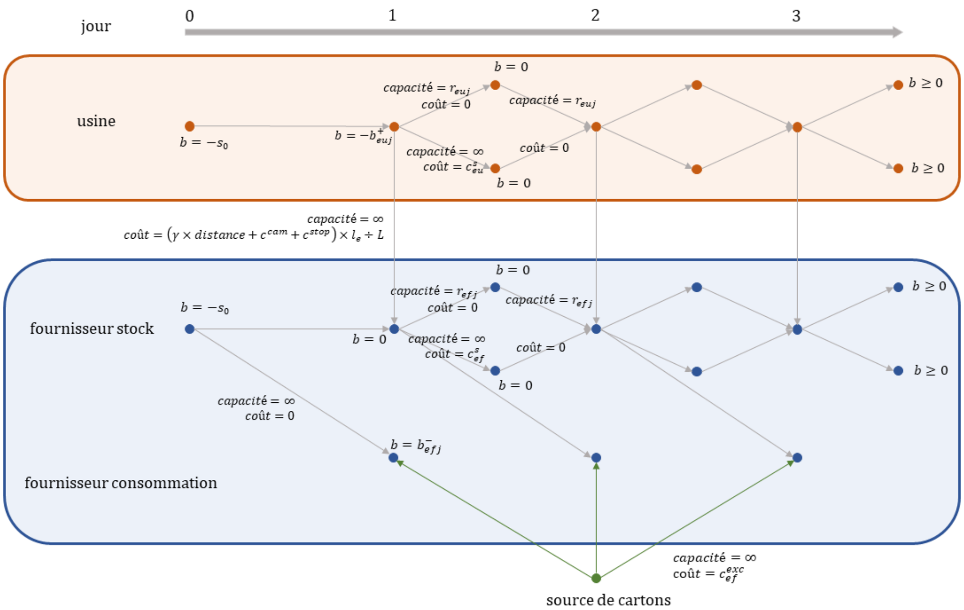

Figure 2 – Graphe de flot de la modélisation du premier sous-problème