

Projet 1 - Problème de production et Eurotunnel

RECHERCHE OPÉRATIONNELLE

Clément RIU et Anne SPITZ

3 janvier 2017

Première partie - Un problème de production

1. Programme linéaire

Pour modéliser le problème, nous utiliserons les notations suivantes :
 T est l'ensemble des instants du problèmes (les semaines de production). T^* est T privé de la semaine 0. \mathcal{R} est l'ensemble des références qu'il est possible de produire.

Ensuite, les données du problèmes sont : $\forall i \in T$ et $\forall r \in \mathcal{R}$, $s_{i,r}$ le stock de la référence r en semaine i et $x_{i,r}$ la quantité produite en semaine i de la référence r . Enfin $d_{i,r}$ est la demande en référence r la semaine i et c_r est le coût du stock de la référence r . Q est la quantité maximale produite chaque semaine et N est la quantité maximale de référence produite chaque semaine.

On cherche à minimiser le coût total (production et stockage). On cherche donc à minimiser

$$\sum_{i \in T^*} \sum_{r \in \mathcal{R}} c_r \times s_{i,r}$$

Nos contraintes sont les suivantes :

- *Contraintes de positivité* : À chaque semaine et pour chaque référence, le stock doit être positif (ou nul). De même, on s'interdit de détruire du stock : on doit donc avoir une production positive (ou nulle) pour chaque référence. On fixe donc

$$\forall i \in T; \forall r \in \mathcal{R}; s_{i,r} \in \mathbb{R}^+$$

$$\forall i \in T; \forall r \in \mathcal{R}; x_{i,r} \in \mathbb{R}^+$$

- À chaque nouvelle semaine, on actualise le stock de la nouvelle semaine avec le stock de la semaine précédente, la production et les références expédiées. On a donc

$$\forall i \in T^*; \forall r \in \mathcal{R}; s_{i,r} = x_{i,r} + s_{i-1,r} - d_{i,r}$$

Sachant que le stock est toujours positif, cette contrainte permet également de s'assurer qu'on satisfait bien la demande chaque semaine.

- D'autre part, on ne peut pas produire plus de N références par semaine, ce qu'on peut traduire par :

$$\forall i \in T^*; \sum_{r \in \mathcal{R}} \mathbf{1}_{x_{i,r} > 0} \leq N$$

- Enfin, il faut satisfaire la capacité de production de l'usine : On ajoute donc la relation :

$$\forall i \in T^*; \sum_{r \in \mathcal{R}} x_{i,r} \leq Q$$

Notre problème se modélise donc par programme linéaire suivant :

$$\min \sum_{i \in T^*} \sum_{r \in \mathcal{R}} c_r \times s_{i,r}$$

sous contraintes :

$$\forall i \in T; \forall r \in \mathcal{R}; s_{i,r} = x_{i,r} + s_{i-1,r} - d_{i,r}$$

$$\forall i \in T^*; \sum_{r \in \mathcal{R}} \mathbf{1}_{x_{i,r} > 0} \leq N$$

$$\forall i \in T^*; \sum_{r \in \mathcal{R}} x_{i,r} \leq Q$$

$$\forall i \in T; \forall r \in \mathcal{R}; s_{i,r} \in \mathbb{R}^+$$

$$\forall i \in T; \forall r \in \mathcal{R}; x_{i,r} \in \mathbb{R}^+$$

Détails d'implémentation :

Pour traduire la condition avec l'indicatrice $\forall i \in T^*; \sum_{r \in \mathcal{R}} \mathbf{1}_{x_{i,r} > 0} \leq N$, on introduit la matrice A , dont l'élément (i,r) vaut 1 si la référence r a été produite pendant la semaine i , et zéro sinon. Il faut donc ajouter une contrainte qui forcera à produire les uniquement les références mentionnées dans la matrice A .

On traduit donc la 2e contrainte par :

$$\forall i \in T^*, \sum_{r \in \mathcal{R}} a_{i,r} \leq N$$

$$\forall r \in \mathcal{R}$$

Par ailleurs, dans l'implémentation, on inverse les lignes et colonnes de d (conformément à la structure des données reçues par mail)

2. Résolution avec GLPK

On récupère les trois sets de données envoyés par mails, et on les place dans les fichiers *Q1_dataset_1.dat*, *Q1_dataset_2.dat* et *Q1_dataset_3.dat*.

Le programme linéaire a été traduit dans GLPK dans le fichier *production.mod*. On peut exécuter tous les programmes d'un coup en lançant *production.sh*.

Première instance

Le programme ne trouve pas de solution réalisable pour la première instance.

En prenant $N' = 3$ (sachant que $N = 2$) et $Q' = Q$, on trouve une solution réalisable, de coût 226. $Q + N$ minore la solution optimale mais n'est pas réalisable, $Q' + N' = Q + N + 1$. Cette instance étant réalisable et s'agissant un problème linéaire en nombres entiers, c'est la solution optimale.

En prenant donc $Q' = Q$ et $N' = N + 1 = 3$, on trouve le plan de production suivant (la matrice se lit avec la ligne r = référence r , colonne i = semaine i , conformément au format des données reçues) :

	0	5	0	6	0	4
	2	0	8	0	0	0
	0	8	0	0	5	0
	3	0	0	7	0	2
$x =$	0	0	7	0	0	2
	0	0	5	0	0	0
	0	5	0	0	8	0
	0	0	0	0	5	0
	5	0	0	7	0	0

avec un coût total de 226.

Si on veut satisfaire précisément cette demande, il faut donc ouvrir une nouvelle ligne de production afin d'être capable de produire une référence supplémentaire dans la semaine (et ainsi augmenter N).

Deuxième instance

On trouve le plan optimal de production suivant (on a toujours $x[r, i]$ la quantité produite de la référence r à la semaine i , conformément au format des données reçues) :

	0	0	2	5	7	4	5	0	0	3
	0	14	1	0	0	0	0	0	0	0
	0	0	14	0	0	0	0	8	0	6
	0	0	0	7	1	2	1	3	3	2
	0	0	0	4	0	2	4	4	5	0
$x =$	15	0	0	0	0	0	0	0	0	0
	0	0	0	0	8	0	0	2	0	0
	0	0	0	0	0	4	0	2	3	0
	0	3	6	0	2	0	4	0	3	0
	0	0	0	4	2	0	6	0	2	5
	8	6	0	0	0	8	0	0	0	4
	0	0	0	3	3	3	3	4	7	3

avec un coût total de 766.

Troisième instance

Le programme renvoie le plan optimal de production suivant (on a toujours $x[r, i]$ la quantité produite de la référence r à la semaine i , conformément au format des données reçues) :

$$x = \begin{pmatrix} 0 & 18 & 0 & 0 & 0 & 8 & 3 & 0 & 0 & 5 \\ 0 & 0 & 0 & 6 & 6 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 9 & 0 & 3 & 2 & 4 & 0 & 3 & 7 \\ 0 & 1 & 10 & 0 & 0 & 3 & 0 & 3 & 5 & 0 \\ 0 & 1 & 2 & 6 & 1 & 2 & 5 & 5 & 5 & 0 \\ 14 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 3 & 1 & 1 & 4 & 0 \\ 1 & 4 & 1 & 8 & 2 & 0 & 0 & 1 & 1 & 2 \\ 0 & 1 & 5 & 0 & 4 & 0 & 6 & 0 & 2 & 5 \\ 0 & 0 & 0 & 4 & 0 & 6 & 6 & 8 & 0 & 4 \\ 0 & 0 & 0 & 3 & 3 & 3 & 2 & 5 & 7 & 3 \end{pmatrix}$$

avec un coût total de 368.

Deuxième partie - Un problème d'ordonnancement de trains et de navettes

Nous avons choisi de formuler le problème sous forme de programme linéaire en nombre entier.

Notation utilisée

Les paramètres du problèmes sont : e , h et p , respectivement le nombre d'eurostars, de HGV et de PAX que l'on souhaite faire passer.

On note $n = e + h + p$ et $n_{mat} = [e, h, p]$.

Enfin, t est le tableau des tampons comme défini dans l'énoncé.

On introduit également la quantité $f_{max} = \frac{3600}{n-1}$, qui correspond à une borne supérieure évidente de la flexibilité.

Enfin, on introduit les variables suivantes :

- T est une matrice $n \times 3$ qui décrit l'ordre des trains qui partent.
Ainsi, le coefficient $T_{i,j}$ vaut 1 si un train de type j est le $i^{\text{ième}}$ mobile à entrer dans le tunnel. Sinon, il vaut 0.
- D est une matrice $n \times 3$ qui donne la date de départ des trains. Là où T est nulle, D est nulle. Sinon, la valeur de $D_{i,j}$ est la date (en secondes) où le train part du quai. (Notons que le premier départ possible est à 1s et le dernier à 3600s au lieu de 0s et 3599s).
- E est une matrice $n \times n$ qui va montrer les moments de départs des eurostars de la première demi-heure. Ainsi, $E_{i,j}$ vaut 1 si le $i^{\text{ième}}$ train à partir est un eurostar, le $j^{\text{ième}}$ train est également un eurostar et qu'ils sont séparés de 1800s. Cela équivaut à $\mathbf{1}_{T_{i,1}=T_{j,1}=1} \cup \mathbf{D}_{j,1}-\mathbf{D}_{i,1}=1800$.
- S est un vecteur à n lignes. Sa $i^{\text{ième}}$ composante vaut 1 si le train partant en $i^{\text{ième}}$ position est un eurostar et que sa date de départ est inférieure à 1800s.
- Finalement, f_{value} est la valeur de la flexibilité.

Représentation linéaire du problème

Notre fonction objectif est donc simplement f_{value} qu'on cherche à maximiser.

- Il n'y a qu'un seul train qui part à chaque horaire : $\forall i \in [n]; \sum_{j=1}^3 T_{i,j} = 1$.
- On envoie le bon nombre de chaque type de train : $\forall j \in [3]; \sum_{i=1}^n T_{i,j} = n_{mat}[j]$.
- On contraint D à être nulle quand T est nulle et sinon, à être entre 1 et 3600 :
 $\forall i \in [n], \forall j \in [3]; T_{i,j} \leq D_{i,j} \leq 3600 * T_{i,j}$.
- Cette contrainte permet de traduire la contrainte d'écart minimal entre deux trains consécutifs :
 Si on considère dans un premier temps les trains qui se suivent dans la même heure, on a une contrainte du type : $D_{k,l} - D_{i,j} \geq t_{j,l} + f_{value}$ avec $k > i$.
 Si les trains se suivent sur deux heures différentes, on modifie simplement de la sorte :
 $D_{k,l} - (D_{i,j} - 3600) \geq t_{j,l} + f_{value}$.
 Enfin, lorsqu'on parcourt D , il y a des cas où la valeur sera nulle, et il ne faut donc pas que cette contrainte s'applique. C'est pourquoi on soustrait au terme de droite la valeur : $(3600 + 480 + f_{max}) \times ((1 - T_{i,j}) + (1 - T_{k,l}))$ qui vaut 0 lorsque $D_{i,j}$ et $D_{k,l}$ sont non nuls, et est plus grand que $t_{j,l} + f_{value}$ sinon.

En ajoutant ces conditions, les indices peuvent librement parcourir $[n]$ et $[3]$.

- $\forall i \in [n-1], \forall k \in [i, n], \forall (j, l) \in [3]^2; D_{k,l} - D_{i,j} \geq t_{j,l} + f_{value} - (3600 + 480 + f_{max}) \times ((1 - T_{i,j}) + (1 - T_{k,l}))$
- $\forall i \in [n-1], \forall k \in [i, n], \forall (j, l) \in [3]^2; D_{k,l} - (D_{i,j} - 3600) \geq t_{j,l} + f_{value} - (3600 + 480 + f_{max}) \times ((1 - T_{i,j}) + (1 - T_{k,l}))$

Intéressons nous maintenant à la contrainte des eurostars *liés* (c'est à dire séparés de 30 minutes exactement).

- Tout d'abord, on s'assure que E ne donne d'informations que sur les eurostars de la première demi-heure : $\forall i \in [n], \forall j \in [i]; E_{i,j} = 0$.
- On s'assure ici qu'il existe bien un, et un seul, eurostar associé à un autre eurostar :
 - $\forall i \in [n-1]; \sum_{j=i+1}^n E_{i,j} = S_i$.
 - $\forall j \in [2, n]; \sum_{i=1}^{j-1} E_{i,j} = T_{i,1} - S_i$.
- Enfin, on fixe l'écart entre deux trains à 30 minutes, s'il s'agit bien de deux eurostars *liés* :
 $\forall i \in [n-1], \forall j \in [i+1, n]; -5400 * (1 - E_{i,j}) \leq D_{j,1} - D_{i,1} - 1800 \leq 1800 * (1 - E_{i,j})$

Notre programme linéaire s'écrit donc de la façon suivante :

$$\begin{aligned}
 & \max \quad f_{value} \\
 & \text{sous contraintes :} \\
 & \forall i \in [n]; \sum_{j=1}^3 T_{i,j} = 1 \\
 & \forall j \in [3]; \sum_{i=1}^n T_{i,j} = n_{mat}[j] \\
 & \forall i \in [n], \forall j \in [3]; T_{i,j} \leq D_{i,j} \leq 3600 * T_{i,j} \\
 & \forall i \in [n-1], \forall k \in [i, n], \forall (j, l) \in [3]^2; \\
 & \quad D_{k,l} - D_{i,j} \geq t_{j,l} + f_{value} - (3600 + 480 + f_{max}) \times ((1 - T_{i,j}) + (1 - T_{k,l})) \\
 & \forall i \in [n-1], \forall k \in [i, n], \forall (j, l) \in [3]^2; \\
 & \quad D_{k,l} - (D_{i,j} - 3600) \geq t_{j,l} + f_{value} - (3600 + 480 + f_{max}) \times ((1 - T_{i,j}) + (1 - T_{k,l})) \\
 & \forall i \in [n], \forall j \in [i]; E_{i,j} = 0 \\
 & \forall i \in [n-1]; \sum_{j=i+1}^n E_{i,j} = S_i \\
 & \forall j \in [2, n]; \sum_{i=1}^{j-1} E_{i,j} = T_{i,1} - S_i \\
 & \forall i \in [n-1], \forall j \in [i+1, n]; -5400 * (1 - E_{i,j}) \leq D_{j,1} - D_{i,1} - 1800 \leq 1800 * (1 - E_{i,j})
 \end{aligned}$$

La traduction de ce programme sur GLPK se trouve dans le fichier *eurotunnel.mod*.

En lançant plusieurs fois notre programme linéaire, celui-ci propose plusieurs grilles horaires, comme par exemple :

	<i>H1</i>	<i>H2</i>	<i>H3</i>	<i>H4</i>	<i>E1</i>	<i>E2</i>	<i>P1</i>
	1	224	447	670	1192	1415	1578
<i>Grille horaire</i> ₁ =	<i>H5</i>	<i>H6</i>	<i>H7</i>	<i>H8</i>	<i>E3</i>	<i>E4</i>	<i>P2</i>
	1801	2024	2247	2470	2993	3215	3378

et

	<i>H1</i>	<i>H2</i>	<i>E1</i>	<i>E2</i>	<i>P1</i>	<i>H3</i>	<i>H4</i>
	1	224	747	970	1132	1355	1578
<i>Grille horaire</i> ₂ =	<i>H5</i>	<i>H6</i>	<i>H7</i>	<i>H8</i>	<i>E3</i>	<i>E4</i>	<i>P2</i>
	1801	2024	2547	2770	2932	3155	3378

Qui ont toutes une flexibilité de 42 secondes. Comme les grilles horaires correspondent à une solution optimale d'un programme linéaire en nombres entiers, la flexibilité trouvée est donc optimale.

2. Le plus de navettes HGV possibles

On augmente h dans *Q2_dataset_1.dat*. Le temps de résolution augmente très rapidement, c'est pourquoi pour déterminer la valeur maximale de h pour laquelle il existe des solutions réalisables, on fixe la flexibilité à zéro (si il n'existe aucune solution de flexibilité nulle, alors il n'existe aucune solution réalisable au problème). À partir de $h = 12$, le solveur indique qu'aucune solution réalisable n'existe : $h = 11$ est donc le maximum de navettes HGV qu'on puisse faire passer pour $e = 4$ et $p = 2$.

On refait donc tourner le programme linéaire pour $h = 11$, et le solveur propose la grille suivante :

Grille horaire =

<i>P1</i>	<i>P2</i>	<i>H1</i>	<i>H2</i>	<i>H3</i>	<i>H4</i>	<i>H5</i>	<i>E1</i>		
120	240	420	600	780	960	1140	1620		
<i>E2</i>	<i>H6</i>	<i>H7</i>	<i>H8</i>	<i>H9</i>	<i>H10</i>	<i>H11</i>	<i>E3</i>	<i>E4</i>	
1800	2040	2220	2400	2580	2760	2940	3420	3600	

qui a une flexibilité nulle. On peut donc faire passer au plus $h = 11$ navettes HGV, mais la grille horaire sera (nécessairement) de flexibilité nulle.