KIRO - 1ère Edition

G. Desforges A. Parmentier S. Richard

16 mai 2018

Air France est un acteur international de premier plan du transport aérien. Ses activités couvrent le transport de personnes, le fret et la maintenance aéronautique. Depuis ses hubs aux aéroports de Paris-Orly et Paris-Charles de Gaulle, la compagnie opère des vols vers *plus de 320 destinations et 118 pays*.

Compte tenu de la taille des réseaux opérés par les compagnies aériennes, celles-ci ont été pionnières dans le développement de la recherche opérationnelle, et le bon usage des outils de cette discipline est aujourd'hui incontournable pour la performance de ces compagnies. Les deux domaines principaux d'application de la recherche opérationnelle pour une compagnie aérienne sont la gestion de la demande et la gestion des opérations.

Ce Hackathon porte sur une des principales problématiques de gestion des opérations : la construction des séquences de vols opérées par les avions.

1 Description du problème

L'objectif du problème est de construire les séquences de vols ou *rotations* opérées par les avions de la compagnie aérienne.

Soit V *l'ensemble de vols* $v \in V$ opérés par une flotte d'avions de la compagnie aérienne sur un horizon temporel donné.

Soit P *l'ensemble des avions* disponibles pour opérer ces vols.

Pour chaque vol $v \in V$ et avion $p \in P$, soit $c_{v,p}$ le coût d'opération du vol v avec l'avion p. Ce coût modélise notamment la consommation de carburant.

Une *correspondance* a est une paire de vols (v_1, v_2) qui peuvent être opérés successivement par le même avion. Pour qu'une paire de vols (v_1, v_2) soit une correspondance, certaines contraintes doivent être satisfaites. Par exemple, v_2 doit partir de l'aéroport où v_1 arrive après l'arrivée de v_1. Ces contraintes étant fastidieuses à vérifier du fait de la gestion des fuseaux horaires, nous vous fournissons directement l'ensemble A des correspondances qui peuvent être réalisées.

Le *graphe des vols* est le graphe orienté acircuitique $D = (V, A)$.

Une *rotation* est un chemin dans le graphe des vols, c'est à dire une séquence de vols $(v_0, \dots, v_k)$ telle que (v_{i-1}, v_i) appartient à A pour tout i dans $\{1, \dots, k\}$. Les rotations sont opérées par les avions $p \in P$.

Certaines opérations de maintenances doivent être réalisées sur les avions au minimum tous les κ jours. Ces opérations requièrent des équipements qui ne sont présents que dans certains aéroports appelés *bases de maintenance*. Une opération de maintenance est réalisable sur une correspondance uniquement si l'avion est dans une base de maintenance et si la durée de la correspondance est suffisante pour permettre de réaliser une opération de maintenance. Nous notons A^{m} l'ensemble des correspondances sur lesquelles une opération de maintenance peut être réalisée. On a $A^{\mathrm{m}} \subsetneq A$. Nous notons t_a le nombre de jours entre le début du premier vol de a et le début du second vol de a pour toute correspondance. Nous affectons un coût γ à la réalisation d'un vol v par un avion qui n'a pas visité une maintenance depuis κ jours.

Nous faisons les hypothèses simplificatrices suivantes :

- chaque avion peut commencer sa rotation par n'importe quel vol (en pratique, l'avion est dans un aéroport et ne peut commencer que par vol depuis cet aéroport) ;
- chaque avion sort de maintenance lorsqu'il commence son premier vol (en pratique, ce n'est pas le cas si l'avion n'est pas dans une base).

Nous notons $v \in r$ l'appartenance du vol v à la rotation r.

Soit $V^{\mathrm{m}}(r)$ l'ensemble des vols de r tels que l'avion réalisant r n'a pas visité un arc de maintenance depuis κ jours. Si $r = (v_0, \dots, v_m)$, on a

$$V^{\mathrm{m}}(r) = \left\{ v \in r \colon \exists i, k \text{ tels que } \left| \begin{array}{l} 0 \leq i < k \leq m \\ v = v_k \\ \forall j \text{ tel que } i \leq j < k, \text{ on a } (v_j, v_{j+1}) \notin A^{\mathrm{m}} \\ \text{ et } \sum_{i \leq j < k} t_j \geq \kappa \end{array} \right. \right\}.$$

Le coût d'opération de la rotation r par l'avion p est

$$c^{\mathrm{r}}(r,p) = \gamma|V^{\mathrm{m}}(r)| + \sum_{v\in r} c_{v,p}.$$

Une *solution* est une collection de rotations $\mathcal{R} = (r_p)_{p\in\{1,\dots,P\}}$ affectées à chaque avion. Nous notons $n(v,\mathcal{R})$ le nombre d'avion réalisant le vol v dans la solution $\mathcal{R}$. Chaque vol doit-être couvert par un et un seul avion. Nous affectons un coût β à chaque avion manquant ou en trop réalisant un vol v.

Le coût d'une solution $(r_p)_{p\in\{1,\dots,P\}}$ est donc

$$c(\mathcal{R}) = \sum_{p=1}^{P} c^{\mathrm{r}}(r_p,p) + \beta\sum_{v}|n(v,\mathcal{R})-1| \quad \text{où} \quad \mathcal{R} = (r_p)_{p\in\{1,\dots,P\}}.$$

L'objectif est de construire une solution $\mathcal{R}$ de coût $c(\mathcal{R})$ minimum.

Remarque 1. Du point de vue de la compagnie aérienne, si deux rotations passent par le même vol, cela veut dire qu'un avion réalise un vol sans passagers, ce qui coûte très cher. Si aucune rotation ne passe par un vol, cela veut dire qu'un autre avion devra être mobilisé spécialement pour ce vol, ce qui coûte aussi très cher et doit être évité.

2 Format des instances

Chaque instance est contenue dans un fichier texte. Ce fichier texte suit une structure précise pour décrire les paramètres et le graphe des vols.

- La première ligne contient les paramètres du problème :
  - `V` : le nombre $|V|$ de vols, donc de nœuds du graphe de vols ;
  - `A` : le nombre $|A|$ d'arcs du graphe de vols ;
  - `P` : la taille $|P|$ de la flotte, soit le nombre d'avions disponibles ;
  - `B` : le coût β par vol réalisé en trop ou non réalisé ;
  - `K` : la fréquence κ à laquelle il faut retourner à une base ;
  - `G` : le coût γ à réaliser un vol sans avoir fait la maintenance requise.
- Les $|V|$ lignes suivantes décrivent un vol $v \in V$ chacune. Elles commencent par `v`, suivi de l'index du vol, suivi de `c`, suivi du coût d'opération du vol par chacun des $|P|$ avions.
- Les $|A|$ lignes suivantes décrivent un arc $a \in A$ du graphe de vols (une correspondance) chacune. Elles contiennent :
  - `a` : index de la correspondance $a = (v_1, v_2)$;
  - `o` : index du vol v_1 ;
  - `d` : index du vol v_2 ;
  - `t` : un booléen égal à 1 si $a \in A^{\mathrm{m}}$;
  - `n` : le nombre de jour t_a entre le début et la fin de la correspondance a.

Les paramètres donnés sur chaque ligne sont séparés par des espaces.

Remarque 2. Les sommets sont fournis selon un ordre topologique : s'il existe un chemin du sommet v_1 au sommet v_2, alors l'index de v_1 est plus petit que l'index de v_2.

Voici un exemple d'instance :

```
V 7 A 8 P 3 B 100000 K 5 G 10000
v 1 c 12 15 10
v 2 c 8 9 10
v 3 c 5 6 7
v 4 c 2 3 4
v 5 c 4 5 6
v 6 c 7 8 9
v 7 c 1 3 2
a 1 o 1 d 3 t 1 n 5
a 2 o 1 d 4 t 1 n 1
a 3 o 2 d 3 t 1 n 4
a 4 o 2 d 4 t 0 n 0
a 5 o 5 d 6 t 1 n 3
a 6 o 5 d 7 t 1 n 4
a 7 o 4 d 6 t 1 n 2
a 8 o 4 d 7 t 1 n 3
```

Dans cet exemple :

- Il y a 3 avions dans la flotte.

— Il y a 7 vols à réaliser. Le vol 1 a par exemple un coût de 12 si l'avion 1 le fait, un coût de 15 si l'avion 2 le fait ou un coût de 10 si l'avion 3 le fait.
— Il y a 14 décisions possibles. Par exemple la décision 1 consiste à faire suivre à un avion le vol 3 après le vol 1, ce qui le fait rester au sol 5 jours et compte comme un arrêt à une base.
— Un avion doit faire une pause à une base avant 5 jours.
— Ne pas faire un vol coûte 100000.
— Avoir un avion ne passant pas par une base en 5 jours coûte 10000.

3 Format des solutions

Les solutions doivent être postées sur le scoreboard au format suivant.

Le ficher contient une ligne par avion, décrit de cette façon
— `p` : index de l'avion (de 1 à $|P|$)
— `v` : séquence des indices des vols opérés par l'avion

Vous trouverez ci-dessous un exemple de fichier solution pour l'instance ci-dessus

```
p 1 a 2 4 7
p 2 a 1 3
p 3 a 5 6
```

Dans cette solution, l'avion 1 réalise les vols 2, 4 et 7, l'avion 2 réalise les vol 1 et 3 et l'avion 3 réalise le vol 5 et 6.

L'avion 1 réalise donc des vols aux coûts 8, 2 et 1, pour un coût total de 11. L'avion 2 vol pour un coût total de 21, l'avion 3 pour un coût total de 15.

4 Comment sera établi le classement

Il y a quatre instances :
- `S.in` une instance d'exemple ;
- `M.in` une petite instance ;
- `L.in` une instance plus grande ;
- `XXL.in` une très grande instance.

Au départ, toutes les instances pour lesquelles vous n'aurez pas envoyé de solution auront un coût pour votre équipe.

Pour chaque taille d'instance, seule la meilleure solution (celle au coût minimal) que votre équipe aura soumise sera comptabilisée.

Le score de votre équipe sera la somme des meilleurs coûts obtenus.

L'équipe ayant le score final le plus faible remportera le Hackathon.

5 Quelques conseils

Six heures, c'est très court. L'équipe qui gagnera sera celle qui arrivera à produire le plus vite des solutions décentes.

1. Partagez vous les tâches.
2. Commencez immédiatement à coder les outils pour parser un fichier d'entrée et créer un fichier de sortie.
3. Comparez β, γ et les $c_{v,p}$: n'hésitez pas à négliger certains aspects de la fonction de coût qui sont difficiles à coder et ont relativement peu d'impact. Les prendre en compte sera la cerise sur le gâteau s'il vous reste du temps à la fin.