Optimisation robuste et en temps réel du revenu d'une compagnie aérienne

Oriana GOYONS[1], Benoit LARDEUX[1], Charles-Antoine ROBELIN[1]

Amadeus SAS Group IT, 485, Route du Pin Montard, 06560 Sophia Antipolis, France
{ogoyons,blardeux,crobelin}@amadeus.com

Mots-Clés : *optimisation robuste, gestion du revenu, programmation linéaire*

1 Introduction

Les compagnies aériennes font face à un problème de produits périssables : chaque vol possédant un nombre limité de place, il faut remplir l'avion au mieux avant son départ. Pour cela, la vente de ces produits s'organise en plusieurs classes de réservation, chacune correspondant à des conditions de ventes (billet remboursable, échangeable...), à un type de siège (affaire ou économique) et à un prix. L'objectif d'une méthode de contrôle des stocks est d'allouer au mieux les ressources afin d'optimiser les revenus : il s'agit donc de savoir, pour une demande de réservation à un temps donné, s'il faut l'accepter ou la refuser.

Dans ce document, nous présentons une nouvelle méthode de contrôle des stocks, robuste face à l'incertitude sur l'estimation de la demande, et appliquée à chaque requête de voyage. Cette méthode permet de calculer le manque à gagner résultant de la vente d'un billet, dans le but de déterminer la disponibilité de ce billet. Le calcul du manque à gagner est obtenu grâce à un modèle robuste consistant à minimiser le regret maximal entre les solutions de différents scénarios générés à partir de l'estimation de la demande. La méthode développée est basée sur la programmation linéaire.

2 Présentation du problème

Considérons un réseau, soit, pour un jour donné, l'ensemble des vols d'une companie aérienne. Celui-ci est constitué de m ressources (pour chaque vol, on associe un ensemble de sièges à une ressource) et de n produits (un produit est une classe de réservation associée à un ou plusieurs vols consécutifs). Chaque produit est donc une combinaison de ressources. La matrice A, appelée matrice d'incidence fait correspondre les produits aux ressources : $a_{i,j}$ est égal à 1 si la ressource i est utilisée par le produit j. Soit c le m-vecteur des capacités restantes : c_i est le nombre de sièges disponibles à la vente pour la ressource i. On génère suivant la distribution de la demande, un nombre S d'estimations différentes de la demande pour chaque produit. On note μ_s le n-vecteur des estimations des demandes restantes sous le scénario $s \in [0; S]$. Sous chaque scenario s, le contrôle optimal est la solution qui maximise les revenus en respectant les contraintes de capacité sur les ressources et les quantités maximales de demande restante estimée sur chaque produit :

$$P(c,s)\left\{\begin{array}{l} V_s^*(c) = \max\limits_x \sum\limits_{j=1}^{n} p_j.\min(x_j, \mu_j^s) \\ \hfill s.t. : \\ \hfill \begin{array}{rcll} \sum_{j=1}^{n} a_{ij} * x_j & \leq & c_i & \forall i \in [1,m]\ (1) \\ 0 & \leq & x_j & \forall j \in [1,n]\ (2) \end{array} \end{array}\right.$$

Soit $x^{*LP}(c,s)$ le contrôle optimal de ce programme. Il correspond pour chaque produit à la quantité que l'on devrait vendre afin d'optimiser le revenu. $V_s(x,c)$ est le revenu généré par le contrôle x, réalisable dans $P(c,s)$. Le but de Robust Dynamic Inventory Control est de choisir la solution qui minimise le regret maximum sous chaque scénario soit l'écart maximum de revenu entre la solution choisie et les solutions optimales sous chaque scénario.

Avec ce contrôle, il nous est possible de calculer le manque à gagner d'un produit j et de savoir s'il est avantageux de le proposer à la vente maintenant. Ce calcul a été proposé par Bertsimas et Popescu [1]. Il a été ici légèrement modifié pour tenir compte du nombre de scénarios générés :

$$\frac{1}{|S|} * \sum_{s\in S}(V_s(x^{*robustP}(c)), c) - \frac{1}{|S|} * \sum_{s\in S}(V_s(x^{*robustP}(c-A_j)), c-A_j)$$

$x^{*robustP}(c)$ est le contrôle optimal courant qui est connu. Le problème est de trouver $x^{*robustP}(c-A_j)$ en peu de temps. Ce contrôle semble difficile à obtenir en un temps acceptable pour une requête de disponibilité, en raison de la grande taille et de la non-linearité du programme précédent. Cependant, des heuristiques dont une présentée ci-dessous peuvent nous permettre de trouver une solution réalisable se rapprochant de l'optimum en quelques dizièmes de secondes.

Un moyen de diminuer le temps de calcul est de réduire l'espace de recherche : nous allons nous contenter de ne regarder que les solutions optimales de chaque scénario. Celles-ci peuvent s'obtenir facilement en linéarisant le programme. Ensuite, $x^{*LP}(c-A_j,s)$ peut s'obtenir rapidement à partir de $x^{*LP}(c,s)$ utilisant l'algorithme dual du simplex. Enfin, il faut calculer le revenu généré sous chaque scénario pour chaque contrôle précédement trouvé et choisir celui qui minimise le regret maximal. On remarquera que plusieurs opérations sont totalement indépendantes les unes des autres et peuvent être réalisées en multithreading. Cette première heuristique peut être complexifiée en élargissant l'espace de recherche aux combinaisons convexes des contrôles optimaux de chaque scénario.

Cette heuristique a été testée dans un environnement de simulation avec des réseaux de grande taille notament un réseau composé de 2 companies aériennes en compétition, 40 aéroports dont 2 hubs et environ 210 vols par jour. Les résultats obtenus avec Robust Dynamic Inventory Control lancé avec des instances allant jusqu'à 20 scénarios, montrent un gain de revenu par rapport aux méthodes existant dans la littérature (Emsrb, Itérative DAVN[2]).

Références

[1] D. Bertsimas et I. Popescu. Revenue Management in a dynamic network environment. *Transportation Science*, 37(3):257-277, 2003.

[2] Talluri and Van Ryzin. The Theory and Practice of Revenue Management. *The Handbook of Airline Economics*, Vinod, New York, 1995.