Recherche locale pour un problème de réapprovisionnement multi-échelon avec approvisionnement multiple

Agathe Métaireau[1,2], Clarisse Dhaenens[2], Nadarajen Veerapen[2], Manuel Davy[1]

[1] Vekia, 16 rue Faidherbe, 59000, Lille, France
[2] Univ. Lille, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000, Lille, France

Mots-clés : *Approvisionnement ; Gestion des stocks ; Recherche locale*

1 Introduction : Contexte et problème étudié

Dans les problèmes de réapprovisionnement, on cherche à calculer les meilleures commandes à passer auprès de fournisseurs afin de satisfaire une demande incertaine tout en minimisant les coûts et les stocks restant en fin de période. Lorsque l'on cherche à optimiser l'approvisionnement d'une chaine logistique, l'approche la plus répandue est de séparer ce système en entités indépendantes pour calculer séparément les commandes à passer. C'est ce qu'on appelle l'approche simple-échelon. Cependant, les entités d'une chaine logistique communiquent entre elles, et peuvent partager des contraintes, qu'une approche simple-échelon ne peut pas prendre en compte. Dans ce cas, on peut chercher à résoudre le problème d'approvisionnement pour toutes les entités du système de façon conjointe. C'est ce qu'on appelle l'approche multi-échelon [1], et c'est l'approche que l'on a choisie pour ce travail, car elle nous permet de mieux prendre en compte certaines contraintes réelles.

Dans cet article, nous travaillons sur un cas d'étude tiré de données réelles. Nous considérons un réseau constitué d'un entrepôt et de plusieurs magasins. L'entrepôt commande auprès de fournisseurs externes, et remplit les commandes des magasins. Les magasins peuvent aussi placer des commandes directement auprès des fournisseurs externes. Nous considérons un système contenant plusieurs produits, et avec des contraintes multi-produits, telles que la capacité de stockage, le coût fixe de commande, et le coût de Franco. Nous sachons que, dans le cas de contraintes multi-produits, il peut être intéressant de placer des commandes plus élevées, afin d'atteindre certains seuils. C'est pourquoi on a un intérêt à regarder plusieurs périodes en avance pour l'optimisation de l'approvisionnement. On considère donc un problème de réapprovisionnement multi-échelon, multi-produit et multi-période.

Nous avons modélisé ce problème sous forme de programme linéaire, et avons assez vite atteint les limites du solver CPLEX avec les données de terrain. En effet, pour les instances les plus complexes que nous avons testées, CPLEX n'arrive pas à trouver une solution initiale dans le temps limite que nous lui avons laissé, à savoir deux heures. C'est pourquoi nous avons cherché une méthode approchée pour résoudre notre problème. Etant donné que nous travaillons avec des variables discrètes et un espace de faisabilité très large, les métaheuristiques sont l'approche la plus adaptée [2].

2 Résolution : Recherche locale multi-opérateur

Comme mentionné précédemment, nous nous orientons vers une métaheuristique pour résoudre le problème considéré. Etant donné que l'on travaille sur un problème assez contraint, on ne peut pas construire une solution rapidement. Nous nous orientons donc vers un algorithme à solution unique, plutôt que vers un algorithme basé sur une population de solutions. La recherche locale est un algorithme qui se prête bien à notre problème.

Nous avons donc cherché à mettre en place une recherche locale pour résoudre le problème considéré. Après avoir testé plusieurs opérateurs de voisinages indépendamment, puis séquentiellement, nous nous sommes rendus compte qu'il était plus intéressant de combiner plusieurs opérateurs afin d'obtenir une solution de meilleure qualité. Nous avons donc mis en place une recherche locale multi-opérateur. Le pseudo-code de cette recherche locale est détaillé par l'algorithme 1.

Algorithme 1 Recherche locale multi-opérateur adaptative

Input : Ensemble d'opérateurs Ops, Critère de terminaison TC, Critère de terminaison local TCL
Calculer les scores initiaux des opérateurs $\mathbb{S}_{Ops}$
Déduire les probabilités de sélection associées $\mathbb{P}_{Ops} = \left(\frac{S_O}{\sum_{O \in Ops} S_0} for\ O\ in\ Ops \right)$
Générer une solution initiale sol^0, Calculer son coût C^0
while not TC **do**
    Choisir un opérateur $O \in Ops$ en fonction des probabilités $\mathbb{P}_{Ops}$
    while not TCL **do**
        Générer un voisin candidat *sol'* avec o à partir de sol^0, Calculer son coût C'
        if $C' < C^0$ **then**
            $sol^0 \leftarrow$ *sol'*, $C^0 \leftarrow C'$
        else
            Augmenter la quantité comparée au critère de terminaison TCL
        end if
    end while
    Calculer le nouveau score de l'opérateur S'_O
    Augmenter la quantité comparée au critère de terminaison TC
    Calculer les nouvelles probabilités $\mathbb{P}_{Ops}$ en tenant compte de S'_o
end while

L'algorithme que nous mettons en place nous permet d'utiliser plusieurs opérateurs pendant la recherche, et va aussi chercher à utiliser les opérateurs les plus pertinents à chaque étape. En effet, nous allons commencer par calculer un score d'efficacité pour chaque opérateur à partir de la solution initiale. Ce score nous permettra de déduire la probabilité de sélection de l'opérateur. Ensuite, à chaque fois que nous appelleront un opérateur pendant la recherche locale, nous allons faire évoluer son score, et donc sa probabilité, en fonction de sa performance immédiate.

Afin de concevoir l'algorithme qui sera le plus adapté à notre cas d'étude, nous devons faire plusieurs choix.

— Nous devons sélectionner le groupe d'opérateurs qu'on va utiliser pour la recherche.
— Nous devons définir comment calculer le score d'efficacité des opérateurs et comment faire évoluer ce score pendant la recherche.
— Nous devons choisir les hyperparamètres de l'algorithme (par exemple, le temps alloué au calcul des scores initiaux ou le critère de terminaison), et donc mettre en place un protocole de réglage de ces paramètres.

Nous présenterons donc comment nous avons effectué ces différents choix de conception de l'algorithme.

Références

[1] Ton de Kok, Christopher Grob, Marco Laumanns, Stefan Minner, Jörg Rambau, and Konrad Schade. A typology and literature review on stochastic multi-echelon inventory models. *European Journal of Operational Research*, 269(3) :955–983, 2018.

[2] Hamed Jalali and Inneke Van Nieuwenhuyse. Simulation optimization in inventory replenishment : a classification. *IIE Transactions*, 47(11) :1217–1235, 2015.