

Résolution du SSCFLP

Dumez Dorian

November 17, 2016

Introduction

Nous allons ici nous intéresser aux problèmes de FLP. Le but étant de résoudre de manière exacte, grâce à un algorithme de branch & bound, le problème de SSCFLP. Le problème de FLP, facility location problem, consiste à trouver où ouvrir des services pour couvrir tous ses clients à moindre frais. La version qui nous intéresse, la single source avec capacité, rajoute des contraintes de capacité et de condition de livraison pour les clients. Le but de cette étude est d'utiliser la version sans capacité et celle sans contrainte de source unique comme relaxation de notre problème pour obtenir des bornes pour notre algorithme.

1 Problèmes étudiés

La première, et plus simple, version de notre problème est le UFLP. C'est à dire la localisation de service sans capacité. On se permet alors de supposer qu'un dépôt peut alimenter un nombre infini de clients, on ne quantifie donc pas la demande des clients.

Le modèle de programmation linéaire est :

$$\begin{aligned} \min z &= \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \\ \text{s.c.} \end{aligned}$$

$$\forall i \in I : \forall j \in J : y_{ij} - x_j \leq 0 \quad (1)$$

$$\forall i \in I : \sum_{j \in J} y_{ij} \geq 1 \quad (2)$$

$$\forall j \in J : x_j \in \{0; 1\} \wedge \forall i \in I : y_{ij} \in \{0; 1\}$$

Les x_j représentent le fait que la facilité j est ouverte et les y_{ij} le fait que le client i est appareillé avec le dépôt j . Les données sont les coûts d'ouvertures f_j et les coûts d'association client-dépôt c_{ij} .

La contrainte 1 exprime le fait que le dépôt j peut alimenter le client i seulement si il est ouvert. La deuxième exprime le fait que tous les clients doivent être couverts.

Dans une deuxième version du problème, CFLP, on prend en compte des capacités et des demandes. C'est à dire que les dépôts ont une capacité limitée et que les clients ont une demande quantifiée. De plus il faut remarquer que tous les dépôts n'ont pas forcément

la même capacité et les clients des quantité de demandes différentes, même si tout cela concerne toujours le même produit. Enfin il faut remarque que l'on interdit pas qu'un client ait sa demande soit fragmenté entre plusieurs facilités.

Le modèle de programmation linéaire est :

$$\begin{aligned} \min z &= \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \\ \text{s.c.} \end{aligned}$$

$$\forall j \in J : \sum_{i \in I} y_{ij} \leq s_j x_j \quad (1)$$

$$\forall i \in I : \sum_{j \in J} y_{ij} = w_i \quad (2)$$

$$\forall j \in J : x_j \in \{0; 1\} \wedge \forall i \in I : y_{ij} \in \mathbb{R}^+$$

Maintenant que des quantités entre en jeux les variables y_{ij} exprime la quantité fournie au client i par le déposé j . Et les données supplémentaires sont les capacités s_j des déposé et les demandes w_i des clients. Il faut aussi noter que les coûts d'associations sont maintenant exprimé en prix par unité transporté.

La contrainte 1 représente alors le fait qu'un déposé ne peut fournir plus que sa capacité, si il n'est pas ouvert sa capacité est nécessairement de 0. Et la seconde force la satisfaction complète de la demande de tous les clients.

La dernière version du problème, celle à laquelle on s'intéresse, le SSCFLP, ajoute la contrainte qu'un client ne peut être servi que par un unique déposé. Le modèle de programmation linéaire est :

$$\begin{aligned} \min z &= \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \\ \text{s.c.} \end{aligned}$$

$$\forall j \in J : \sum_{i \in I} y_{ij} w_{ij} \leq s_j x_j \quad (1)$$

$$\forall i \in I : \sum_{j \in J} y_{ij} = 1 \quad (2)$$

$$\forall j \in J : x_j \in \{0; 1\} \wedge \forall i \in I : y_{ij} \in \{0; 1\}$$

Maintenant que les quantités demandé par les clients ne peuvent plus être fractionné des booléen suffise de nouveau pour les y_{ij} et exprime de nouveau l'association du client i a la facilité j . De plus aucune donnée supplémentaire n'est nécessaire.

La contrainte 1 exprime toujours le fait qu'un déposé ne peut livrer plus que sa capacité. Et la contrainte 2 impose le fait que chaque client doit être livré par un unique déposé.

On remarque alors que le programme de l'UFLP peut être re-écrit si les coûts d'association sont non-négatif :

$$\min z = \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij}$$

s.c.

$$\forall i \in I : \forall j \in J : y_{ij} - x_j \leq 0 \quad (1)$$

$$\forall i \in I : \sum_{j \in J} y_{ij} = 1 \quad (2)$$

$$\forall j \in J : x_j \in \{0; 1\} \wedge \forall i \in I : y_{ij} \in \{0; 1\}$$

En effet, la fonction d'objectif étant en minimisation, la solution optimale ne couvrira jamais, sauf dégénérescence causé par coûts d'appareillages nul, un client 2 fois. On peut alors dire que la contrainte est satisfaite si et seulement si chaque client est couvert une unique fois.

De plus, on peut re-écrire la contrainte qui force un dépôt utilisé à être ouvert pour faire apparaître plus clairement la relaxation du SSCFLP en UFLP :

$$\min z = \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij}$$

s.c.

$$\forall j \in J : \sum_{i \in I} y_{ij} w_{ij} \leq M x_j \quad (1)$$

$$\forall i \in I : \sum_{j \in J} y_{ij} = 1 \quad (2)$$

$$\forall j \in J : x_j \in \{0; 1\} \wedge \forall i \in I : y_{ij} \in \{0; 1\}$$

En effet l'usage d'un M (une valeur devant laquelle toute demande est négligeable) permet d'oublier la contrainte de capacité. La relaxation du SSCFLP en UFLP s'effectue donc en relaxant la première contrainte. La solution obtenue avec ce programme respectera la condition de source unique mais pas celles des capacités.

On remarque aussi que l'on peut aussi l'écrire sous une autre forme, qui rend évident le fait que le CFLP est une relaxation du SSCFLP :

$$\min z = \sum_{j \in J} f_j x_j + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} w_{ij}$$

s.c.

$$\forall j \in J : \sum_{i \in I} y_{ij} w_{ij} \leq s_j x_j \quad (1)$$

$$\forall i \in I : \sum_{j \in J} y_{ij} = 1 \quad (2)$$

$$\forall j \in J : x_j \in \{0; 1\} \wedge \forall i \in I : y_{ij} \in [0; 1]$$

Les y_{ij} représente alors la part de la demande de i qui est assumé par la facilité j . On remarque alors que si l'on prend le programme linéaire du SSCFLP et qu'on y exprime les coûts d'associations en prix par unité alors on a exactement le même programme à l'exception de la relaxation linéaire des y_{ij} . La solution obtenue avec ce programme respectera les contraintes de capacités mais pas celle de source unique.

2 Instances choisies

Toutes les instances viennent du site <http://www-eio.upc.es/elena/sscplp/index.html>.

La première instance, p1, est une petite instance normale, elle ne comprend que 20 clients et 10 possibilités de facilités. Les coûts d'ouverture sont relativement grand par rapport aux coûts de connexions et les capacités sont aussi grandes par rapport aux demandes.

L'instance p2, petite, possède des coût de connexions élevé par rapport aux coûts d'ouvertures.

L'instance p3 est aussi de petite taille mais présente des facilités avec des coûts d'ouvertures très élevés.

L'instance p6, toujours petite, mais nous propose des facilités peu chère mais avec de petites capacités par rapport aux demandes.

Suivant le même schéma on effectue une montée en charge avec les instances p7(normale), p9(petite et pas chère), p10(facilité chères) et p13(coûts de connexions élevé). En effet celles ci on 30 clients pour 15 possibilités de facilités.

Toujours selon le même principe on passe à des instances à 50 clients et 20 sites possibles. On utilise les instances p26(normale), p33(coûts d'ouvertures élevés), p31(facilités peux chères) et p30 (coûts de connexions élevés).

Bibliographie

Sources des instances : <http://www-eio.upc.es/elena/sscplp/index.html>.