Sup Galilée - INFOA2 - Université Sorbonne Paris Nord
OLOC

Mini-projet d' Optimisation Combinatoire

Ce projet est à rendre avec un mini-rapport,

1ère séance du 17 mai 2024
2ème séance le 20 mai 2024

Lors de la 2ème séance : soutenance de pré-rendu :
vous devez témoigner d'une avancée importante sur le projet ce jour-là.

A rendre en monôme ou en binôme **avant le 5 juin 23h59.**

Placement d'antennes pour un opérateur téléphonique

On considère dans ce projet la résolution du problème de placement d'antennes en utilisant des techniques de résolution heuristique et exacte.

1 Définition du problème de p-centre

Un nouvel opérateur téléphoniques veut couvrir une zone géographique en installant de nouvelles antennes de manière à couvrir tout un territoire avec la meilleure couverture possible.
On modèlise son problème comme étant le problème dit du p-centre qui peut se définir de la façon suivante :

Entrée :	- un nuage de n points sur un plan dont on connait les coordonnées
	- on peut déduire les distances d_{ij} entre deux points i et j en calculant la distance euclidienne entre les points.
	- un nombre p (nombre maximal d'antennes disponibles)
Sortie :	Placer au plus p antennes parmi les n points du plan
Objectif :	de manière à minimiser le rayon de la solution, défini comme la distance maximum entre un point et son antenne la plus proche.

Ainsi, ce problème revient à positionner au mieux p antennes sur des emplacements prédéfinis avec l'objectif que le point le plus éloigné d'une antenne, soit le moins éloigné possible.
Le problème du p-centre a été démontré NP-difficile.

2 Manipulation des instances

Nous allons travailler sur des instances créées à partir des coordonnées (longitude, latitude) des villes française. Vous trouverez les caractéristiques de ces villes dans le fichier villes_france.csv. Ce fichier contient pour chaque commune numérotée de 1 à 36830, son département, son nom (dans un format simplifié), sa population au 1er janvier 2010 et ses coordonnées.
Plusieurs instances de tailles diverses pour le problème sont données au format .flp suivant :

```
3
0 codepostal0 x0 y0  f0
1 codepostal1 x1 y1  f1
2 codepostal2 x2 y2  f2
```

où 3 est le nombre de villes,
et chaque ligne code une ville (voir fichier villes_france.csv)

- codepostal vous permet de retrouver le nom de la ville si vous le souhaitez
- les coordonnées x,y
- un nombre f : Nous n'utiliserons pas ce nombre f pour ce projet : vous pouvez ne pas le stocker par la suite.

Le nombre p est une donnée du problème qui n'est fourni dans les instances : en effet, vous pourrez, pour un même nuage de points, résoudre l'instance pour différentes valeurs de p

Question 1 *Récupérez les instances et le code permettant de lire les instances : on s'en servira pour les trois parties suivantes (méthodes exactes, arrondi et heuristiques).*

Question 2 *Réaliser une visualisation de ces instances et surtout des solutions.*
(Notez qu'une première méthode très simple pour obtenir une solution est de tirer p points au hasard !)

3 Méthode exacte

On considère le programme linéaire en nombres entiers (P) suivant utilisant les variables :
x_{jj} égale à 1 si et seulement une antenne est placée au sommets j de l'instances ;
x_{ij} égale à 1 si et seulement un point i est affecté à l'antenne j.

$$Min \quad z$$

$$\sum_{j=1}^{n} x_{jj} \leq p \tag{1}$$

$$\sum_{j=1}^{n} x_{jj} = 1 \quad \forall i \in \{1,\dots,n\} \tag{2}$$

$$x_{ij} \leq x_{jj} \quad \forall i \in \{1,\dots,n\}, \forall j \in \{1,\dots,n\} \setminus \{i\} \tag{3}$$

$$\sum_{j=1}^{n} d_{ij} x_{ij} \leq z \quad \forall i \in \{1,\dots,n\} \tag{4}$$

$$z \geq 0$$

$$x_{ij} \in \{0,1\} \quad \forall i \in \{1,\dots,n\}, \forall j \in \{1,\dots,n\}$$

On peut remarquer qu'elle est équivalente au problème du p-centre.
La cte (1) implique qu'il y a au plus p antennes.
La cte (2) implique que tout point i du plan est soit une antenne, soit affectée à une antenne.
La cte (3) indique que si un point i est affecté à un point j alors j porte une antenne.
La cte (4) insique que le point i a pour distance a son antenne $\sum_{j=1}^{n} d_{ij} x_{ij}$ (car un seul x_{ij} est à 1) : ainsi cette distance est majorée par z.
On cherche à minimiser z qui est supérieure à toutes les distances entre un point et son antenne : l'objectif est bien de minimiser le rayon.

Question 3 *En utilisant un solveur (Cplex par exemple ou Gplk), implémenter une méthode permettant de résoudre des instances du problème.*

Question 4 *Jusqu'à quelle taille peut-on résoudre ces instances ? (au besoin vous pouvez prendre une partie réduite d'une instance fournie)*

Exemple de déclaration variables à doubles indices

```
m = Model(CPLEX.Optimizer)

#Variable binaire à doubles indices
@variable(m, x[1:G.nb_points, 1:G.nb_points], Bin)

#Variable entière à simples indices
@variable(m, u[1:G.nb_points], Int)

#fonction objective à somme double
@objective(m, Min, sum((sum(c[i, j] * x[i, j]) for j = 1:G.nb_points)
                                                  for i = 1:G.nb_points ) )

# contraite pour dire "une valeur x_ij à 1 sur tous les j possibles, à i fixé"
for i in 1:G.nb_points
    @constraint(m, (sum(x[i, j] for j in 1:G.nb_points))== 1)
end
```

4 Méthodes gloutonnes et méta-heuristiques

Lorsque les instances sont de dimensions trop importantes, les solveurs entiers nécessitent un temps trop important pour être utilisable pour le problème. On se propose de développer une méthode heuristique pour le résoudre.

Question 5 *En utilisant un langage de programmation de votre choix, vous développerez plusieurs méthodes de recherche gloutonnes.*
On rappelle qu'une méthode gloutonne est une méthode qui construite une solution réalisable en prenant itérativement des décisions partielles sur l'instance sans les remettre en question. Vous pouvez proposer une première méthode très simple, comme par exemple basée sur le hasard ; ainsi qu'une deuxième (ou d'autres encore) qui s'appuient sur les poids et/ou la nature géographie des instances.]

Question 6 *En utilisant la relaxation linéaire du programme (P) vous obtenez une solution x fractionnaire : utiliser cette valeur en l'arrondissant à une solution entière heuristiquement : le but est d'obtenir une deuxième heuristique gloutonne.*

Question 7 *Coder une méta-heuristique (descente stochastique, recuit simulé, méthode tabou...) pour le problème.*
Une méthode facile à coder sont la descente stochastique itérée qui consiste à relancer plusieurs fois une descente stochastique initialisée à partir d'une solution initiale générée alétatoirement.

5 Comparaison expérimentale

Question 8 *En utilisant la relaxation linéaire des formulations entières de la partie précédente, on obtient une borne inférieure z_b sur la valeur optimale entière. A tout moment de votre méthode heuristique, on peut considérer la meilleure solution réalisable rencontrée $z_{bestsol}$. On peut donc considérer tout au long du déroulement de l'algorithme l'augmentation de la borne inférieure et la réduction de la borne supérieure. On mesure cet écart en tant qu'écart relatif appelé souvent gap :*

$$gap = \frac{z_{bestsol} - z_b}{z_{bestsol}}.$$

Ainsi, dans le pire des cas, la borne supérieure est égale à z_{opt} : on a ainsi $(1 + gap)z_{bestsol} = z_{opt}$ ce qui signifie qu'au pire $z_{bestsol}$ est à $gap\%$ de l'optimum.
Pour la question suivante, donner une analyse expérimentale des capacités de votre heuristique en utilisant la mesure de gap.

Question 9 *Vous devez comparer expérimentalement la validité de votre code sur des instances de tailles réduites*
- valider que vous obtenez une solution réalisable
- visualiser vos solutions

Question 10 *Votre mini-rapport doit comporter une comparaison des différentes méthodes exactes et approchées.*
En répondant aux différentes questions suivantes :
- taille maximale des instances pouvant être résolues en moins de 5 minutes
- qualité des solutions obtenues (écart par rapport à la solution optimale quand vous l'avez, écart à la meilleure borne connue si possible)

Les comparaisons doivent porter sur un jeux d'essai de taille suffisante.