Projet M1 Androide
Bornes inférieures pour le Partitionnement des Graphes

Encadrants : Viet Hung Nguyen
Étudiants : Lucas Berterottière et Marc-Vincent Pereira

12 février 2018

Nous considérons le problème de Partionnement de Graphe dans sa version la plus élémentaire suivante. Soit $G = (V, E)$ un graphe non-orienté connexe où $V = \{1, \dots, n\}$ et $m = |E|$. On suppose que G est creux, i.e. $m = O(n)$ et les arêtes dans E sont pondérées par un vecteur $c \in \mathbb{R}^E$. Nous voulons partitionner l'ensemble V des sommets en des sous-ensembles (ou clusters) $V_1, V_2, \dots, V_k$ de manière à minimiser $\sum_{\substack{ij \in E \\ i \in V_{k_i}, j \in V_{k_j}, k_i \neq k_j}} c_{ij}$, i.e. le poids total des arêtes dans l'inter-clusters. De façon équivalente, on peut maximier le poids total des arêtes dans l'intra-cluster, i.e. maximiser $\sum_{\substack{ij \in E \\ i,j \in V_p, 1 \leq p \leq k}} c_{ij}$.

À chaque arête ij, on associe une variable $x_{ij} \in \{0, 1\}$:

$$x_{ij} = \begin{cases} 1 & \text{si } ij \text{ appartient à l'inter-cluster,} \\ 0 & \text{sinon.} \end{cases}$$

Soit $\mathcal{C}$ l'ensemble des cycle sans corde de G (chordless cycle en anglais). Soit $F \subset E$ and posons $x(F) = \sum_{e \in F} x_e$. Les solutions du programme linéaire en nombres entiers suivant représentent les partitions de V (i.e. si $x_{ij} = 0$ alors i et j appartiennent tous les deux à un même cluster V_p où $1 \leq p \leq k$:

$$x(F) - x(C \setminus F) \leq |F| - 1 \text{ pour tout } C \in \mathcal{C} \text{ et pour tout } F \subset C \text{ t.q. } |F| \text{ impaire} \quad (1)$$

$$x_e \in \{0, 1\} \text{ pour tout } e \in E. \quad (2)$$

L'inégalité (1) est appelée *l'inégalité de cycle*. Les inégalités de cycles sont toutes nécessaires quand le nombre de clusters k est exactement égal à 2. Mais quand ce dernier n'est pas fixée ou borné à une constante, les inégalités de cycle avec les sous ensemble F t.q. $|F| = 1$ (i.e. celles avec le second membre 0, on les appelle les inégalités de cycle homogènes) suffisent pour modéliser les partitions.
Dans ce projet, nous considérons le problème du k-partitionnnement où le nombre de clusters dans la partition est fixé à une constante $k \geq 2$. Il faudrait donc ajouter des variables pour

modéliser cette propriété. On va utiliser les modèles dans [1], [2] où à chaque sommet i, on associe une variable x_i :

$$x_i = \begin{cases} 1 & \text{si } i \text{ est le plus petit sommet (en terme d'indice) dans son cluster,} \\ 0 & \text{sinon.} \end{cases}$$

Le programme linéaire 0/1 (PLNE) du k-partitionnement est le suivant.

$$\min \sum_{ij \in E} c_{ij} x_{ij}$$

$$x_e - x(C \setminus \{e\}) \leq 0 \text{ pour tout } C \in \mathcal{C} \text{ et pour tout } e \in C \tag{3}$$

$$x_j \leq x_{ij} \text{ pour tout } 1 \leq i < j \leq n \text{ et } ij \in E \tag{4}$$

$$x_j + d_{ij} - 1 \geq \sum_{\substack{1 \leq i < j \\ ij \in E}} x_{ij} \text{ pour tout } j \in V \tag{5}$$

$$\sum_{i=1}^{n} x_i = k \tag{6}$$

$$x_e \in \{0, 1\} \text{ pour tout } e \in E$$

$$0 \leq x_i \leq 1 \text{ pour tout } i \in V \tag{7}$$

Contraintes (3) sont les inéqualités de cycle homogènes. Contraintes (4) disent que si $x_{ij} = 0$, i.e. j est dans le même cluster que i avec $i < j$ alors $x_j = 0$. Dans les contraintes (5), d_{ij} est une constante qui est égale au nombre d'arêtes $ij \in E$ t.q. $1 \leq i \leq j-1$. Ces contraintes disent que si $\sum_{\substack{1 \leq i < j \\ ij \in E}} x_{ij} = d_{ij}$ i.e. cette somme atteint son maximum et aucun sommet plus petit que j est dans le même cluster que j alors $x_j = 1$, i.e. j est le plus petit sommet dans son cluster. Contrainte (6) dit tout simplement que le nombre de clusters dans la partition est exactement k.
La relaxation linéaire (PL) du k-partionnement est évidemment comme suit.

$$\min \sum_{ij \in E} c_{ij} x_{ij}$$

$$x_e - x(C \setminus \{e\}) \leq 0 \text{ pour tout } C \in \mathcal{C} \text{ et pour tout } e \in C$$

$$x_j \leq x_{ij} \text{ pour tout } 1 \leq i < j \leq n \text{ et } ij \in E$$

$$x_j + d_{ij} - 1 \geq \sum_{\substack{1 \leq i < j \\ ij \in E}} x_{ij} \text{ pour tout } j \in V$$

$$\sum_{i=1}^{n} x_i = k$$

$$0 \leq x_e \leq 1 \text{ pour tout } e \in E$$

$$0 \leq x_i \leq 1 \text{ pour tout } i \in V \tag{8}$$

Une solution optimale de (PL) donne donc une borne inférieure du problème de k-partionnement. Toutefois, il est difficile de résoudre directement (PL) à cause du nombre en général exponentiel de constraintes (3). Ceci du au fait que le nombre de cycles sans corde d'un graphe peut être

exponentiel. De plus, résoudre (PL) itérativement par plans coupants en utilisant un algorithme de séparation pour (3) peut prendre énormement de temps. Afin de trouver une solution de compromis qui permet d'obtenir une borne inférieure de bonne qualité en un temps raisonnable, dans ce projet, on essaie d'utiliser seulement un sous-ensemble $\bar{\mathcal{C}}$ des contraintes (3) afin de rendre le programme linéaire directement soluble. Cet ensemble $\bar{\mathcal{C}}$ devrait satisfaire deux critères suivants :

- $|\bar{\mathcal{C}}|$ doit être linéaire en terme de n.
- La longueur maximale des cycles dans $\bar{\mathcal{C}}$ ne peut pas dépasser une constante L donnée.
- Le sous-ensemble des contraintes (3) associé à $\bar{\mathcal{C}}$ est une bonne relaxation de ces contraintes.

Appelons (RPL) le programme linéaire obtenu en remplaçant les contraintes (3) dans (PL) par le sous-ensemble des contraintes associé à $\bar{\mathcal{C}}$. On s'intéresse aux bornes inférieures que puisse donner (RPL) au problème du k-partitionnement dans les instances de graphes creux, i.e. $m = O(n)$. Dans ce cas, le nombre total des contraintes dans (RPL) est en $O(n)$ et par conséquent, (RPL) pourrait être directement soluble par un solveur linéaire de type Gurobi ou CPLEX.

Le but du projet est d'expérimenter des classes particulières des sous-ensembles $\bar{\mathcal{C}}$.

Pour une première approche, on peut prendre des sous-ensembles d'une base de cycles de G comme $\bar{\mathcal{C}}$. Pour rappel, un cycle peut être représenté par son vecteur caractéristique dans $\mathbb{R}^E$. On considère *l'espace des cycles* le sous-espace engendré par des vecteurs caractériques des cycles en corps de Galois $GF(2)^n$ (c.à.d des combinaisons linéaires des vecteurs en arithmétique modulo 2). Une base de cycles d'un graphe est un sous-ensembles des cycles tel que les vecteurs caractéristiques associés forment une base de l'espace des cycles. Le nombre de cycles dans une base est égal à $m - n + 1$ et donc en $O(n)$ quand G est creux. Ayant fixé la constante L, on peut piocher que les cycles de longueur $\leq L$ dans la base pour former $\bar{\mathcal{C}}$.

Cahiers des charges (à perfectionner et compléter) :

- Familiser avec un des solveurs (Gurobi ou CPLEX) et la bibliothèque open-source en langage Python *networkx* sur les graphes (networkx possède des routines pour générer des graphes creux aléatoires, générer une base des cycles, divers routines de plus courts chemins...etc).
- Coder une routine de générateur des instances creuse pour le problème du k-partionnement en utilisant networkx.
- Coder une routine pour construire $\bar{\mathcal{C}}$ avec L et G comme paramètres en utilisant networkx.
- Coder un programme python qui résout (RPL) en utilisant les routines ci-dessus et un solveur de PL comme Gurobi ou CPLEX.
- Coder une routine de séparation pour la contrainte de cycle (3). Vous aurez besoin des routines de plus courts chemins de networkx.
- Coder un programme python qui résout (PLNE). Vous aurez besoin de faire un algorithme branch-and-cut avec la contrainte de cycle (3) en "lazy constraints" en utilisant la routine de séparation précédente.
- Faire une étude de comparaison sur la qualités des bornes données par différents sous-ensembles $\bar{\mathcal{C}}$ avec différentes valeurs de n.
- Faire une étude bibliographique sur le calcul des bornes inférieures pour des variants du partionnnement des graphes, pas seulement pour le k-partitionnement.

Références

[1] Zacharie Ales, Arnaud Knippel, and Alexandre Pauchet. Polyhedral combinatorics of the K-partitioning problem with representative variables. *Discrete Appl. Math.*, 211 :1–14, Oct 2016.

[2] Dang Phuong Nguyen, Michel Minoux, Viet Hung Nguyen, Thanh Hai Nguyen, and Renaud Sirdey. Improved compact formulations for a wide class of graph partitioning problems in sparse graphs. *Discrete Optimization*, 25 :175–188, Aug 2017.