Projet M1 Androide
Cahier des charges

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

27 mai 2018

1 Sujet du projet

Le projet porte sur le k-partitionnement ; il nous ai demandé de coder des algorithmes permettant de calculer une solution exacte pour ce problème sur des instances de petite taille ainsi que des algorithme permettant de trouver une bonne borne inférieure pour des instances de graphes de plus grandes taille(en termes de nœuds).
Le k-partitionnement est un problème étudié pour ses applications dans divers milieux comme l'affectation de tâches pour des systèmes multi-processeurs. Il consiste à diviser un graphe en k parties, Soit $V1, V2.....Vk$ des ensembles appelés clusters (ou partition en français),soit G=(V,E) avec V l'ensemble des sommets et E l'ensemble des arêtes ,on doit répartir les sommets du graphes G dans ces clusters tel que $\forall x \in V, x \in Vi$et $\forall j \neq i \notin Vj$. Le problème du k-partitionnement peut avoir plusieurs but, dans notre cas nous devons minimiser la somme des poids des arêtes appartenant à l'inter-cluster (les arrêtes qui ont leurs deux sommets dans des clusters différents). Pour ce faire , le programme linéaire suivant nous est donné :
À chaque arête ij, on associe une variable $x_{ij} \in \{0, 1\}$:

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

à chaque sommet i, on associe une variable x_i :

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

C est l'ensemble des cycles sans cordes de notre graphe. Le programme linéaire 0/1 (PLNE) du k-partitionnement est le suivant.

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

$$x_e - x(C \setminus \{e\}) \leq 0 \text{ pour tout } C \in \mathcal{C} \text{ et pour tout } e \in C \quad (1)$$

$$x_j \leq x_{ij} \text{ pour tout } 1 \leq i < j \leq n \text{ et } ij \in E \quad (2)$$

$$x_j + d_{ij} - 1 \geq \sum_{\substack{1 \leq i < j \\ ij \in E}} x_{ij} \text{ pour tout } j \in V \quad (3)$$

$$\sum_{i=1}^{n} x_i = k \quad (4)$$

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

$$0 \leq x_i \leq 1 \text{ pour tout } i \in V \quad (5)$$

Il nous ai demandé de codé ce programme et éventuellement si il y a des contraintes à rajouter pour pouvoir renforcer le code ,améliorer sa rapidité ,elles doivent être implémentées.

D'après le document que nous a donné notre encadrant, ce programme ne sera pas suffisant pour résoudre des instances de graphes car il peut y avoir un nombre exponentielle de cycle, il faudra donc implémenter une routine permettant de résoudre le problème en séparant ces contraintes, et plus tard coder des algorithmes permettant de trouver des bornes inférieures pour les graphes de plus grande taille.

2 Logiciel utilisé

Dans le cadre de notre projet, le code devra être écrit en python , pour le solveur , nous avons le choix entre Gurobi et CPLEX , comme nous avions déjà utilisé Gurobi en MOGPL et qu'il est gratuit, nous allons utiliser celui-ci.
En python il nous ai demandé de nous documenter et d'utiliser la bibliothèque Networkx pour tout ce qui concerne les générations de graphes, networkx possède également des routines permettant de calculer des plus court chemin dans un graphe.

3 Plan à suivre

Voici un plan à suivre qui nous a été fourni et que l'on a complété :

- Familiser avec un des solveurs (Gurobi ou CPLEX) et la bibliothèque open-source en langage Python *networkx* sur les graphes .
- Coder une routine de génération des instances creuse et connexe pour le problème du k-partitionnement en utilisant networkx, les poids des arrêtes des graphes seront initialisé avec des distributions de loi de probabilité, on peut soit utiliser une distribution gaussienne, soit uniforme.
- Coder une routine pour construire $\bar{\mathcal{C}}$ avec L et G comme paramètres en utilisant networkx, $\bar{\mathcal{C}}$ est un sous ensemble des cycles utilisé pour pouvoir diminuer le nombre de contraintes sur les cycles(on en prend que une partie), L est la longueur maximale d'un cycle appartenant à $\bar{\mathcal{C}}$.
- Coder un programme python qui résout (RPL) le programme relaxé en utilisant les routines ci-dessus et un solveur de PL comme Gurobi ou CPLEX.
- Coder une routine de séparation pour les contraintes de cycle , les routines de plus court chemin seront utiles pour cette partie.
- Coder un programme python qui résout (PLNE) avec 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. Il est demandé de trouver une valeur de L ,avec laquelle on construit notre sous ensemble de cycle, qui permet de trouver un résultat assez rapidement tout en donnant une bonne approximation de la solution réelle.
- Faire une étude bibliographique sur le calcul des bornes inférieures pour des variants du partionnnement des graphes, pas seulement pour le k-partitionnement. Cette partie est à faire si nous avons du temps à la fin de notre projet mais les autres sont à privilégier avant celle-ci