Algorithmes efficaces via la Programmation Linéaire

Mod 4.4 - Recherche Opérationelle

Le but de ce BE est de manipuler un peu les Programmes Linéaires à l'aide de Python et de "coder" un solveur de Programme Linéaire en nombre entiers. Dans une première partie on recodera le problème de "couplage". Et on tentera de comprendre et d'interpréter son dual. Dans une seconde partie, on codera tout ça en Python et on vérifiera (et interprètera) certains résultats que l'on obtiendra. Les questions (*) sont optionnelles. Le nombre (*) dénote la complexité de la question.

1 Couplages, couvertures et Programme Linéaire

Introduction

Considérons le problème suivant. Soit $G = (V, E)$ un graphe. On note n le nombre de sommets de G et m son nombre d'arêtes. Un *couplage* un ensemble d'arêtes X deux à deux sommets-disjointes dans le graphe (si $e = (u, v)$ et $e' = (x, y)$ sont dans X alors $x \neq u, v$ et $y \neq u, v$). Le but du BE est de trouver est de trouver un couplage de taille maximum dans un graphe G.

Un couplage X est dit *parfait* si tous les sommets du graphe appartiennent à (exactement) une arête de X. La recherche de couplage parfait est un problème important, par exemple dans le couplage (création de paire) dans un graphe social. Les arêtes représentent alors les affinités / complémentarités entre individus et le but est que chaque personne soit associé avec quelqu'un avec qui il s'entend bien.

La recherche d'un couplage de taille maximale joue un rôle central en informatique (en particulier dans les graphes bipartis). Que ce soit en ordonnancement (affectations de tâches à des processus), en optimisation (affectations qui maximisent le bien-être global) ou en sociologie et économie (étude de la stabilité d'une structure). Par exemple, si on a un centraliseur dont le rôle est de distribuer les tâches aux différents processus, on désire affecter les tâches à des processeurs qui sauront les réaliser efficacement. De plus on veut perdre le minimum de temps, et donc affecter le maximum de tâches à chaque étape.

Donnons une application particulière du problème de couplage qui vous a certainement concerné: l'affectaion d'étudiants dans les écoles. L'algorithme "Parcours Sup" -dont on a beaucoup parlé ces derniers mois- est un algorithme basé sur la recherche d'un meilleur couplage dans un graphe. Son prédécesseur, APB, était lui aussi basé sur un algorithme similaire. Dans ce cas, on a deux types de sommets: les sommets "écoles" et les sommets "étudiants". Chaque école propose un ensemble d'étudiants qui l'intéressent (en fait, dans Parcours Sup, chaque école classe les étudiants). Et chaque étudiant propose un ensemble d'écoles (dans un monde parfait, il faudrait aussi que chaque étudiant classes les écoles, l'algorithme convergerait plus vite et les gens sauraient avec le 10 Septembre où il se retrouve... Mais c'est un autre problème). Notons A les écoles et B les étudiants. On crée une arête (a, b) entre $a \in A$ et $b \in B$ si et seulement si a est intéressé par b et b est intéressé par a. Pour chaque école

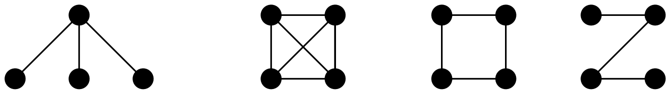

Figure 1: Une clique, une étoile, un chemin et un cycle

a dans A, on a une capacité maximum c_a. Le but est trouver un maximum de paires (a, b) deux à deux disjointes telles que (i) chaque étudiant b est associé à au plus une école; (ii) chaque école a est dans au plus c_a couples. Si $c_a = 1$ pour toutes les écoles, alors il s'agit d'un couplage ! En général, ce n'est bien sur par le cas... Mais il existe une astuce simple pour se ramener au cas où $c_a = 1$ (en "dupliquant" toutes les sommets correspondants aux écoles...). Notez que dans ce cas là, le graphe est *biparti*. (Nous reviondrons plus tard aux graphes bipartis).

Le problème de couplage maximum consiste, étant donné un graphe à trouver un couplage de cardinalité maximum.

Question 1. Quelle est la taille d'un couplage maximum dans une clique (un graphe avec toutes les arêtes) à n sommets? Un cycle à n sommets? Un chemin à n sommets? Une étoile à n branches? (voir Figure 1)

Question 2. Montrer que le problème du Couplage Maximum peut s'exprimer comme un Programme Linéaire en Nombre Entier (PLNE).
(Écrire seulement les contraintes sous la forme d'inégalités pour l'instant).

La *matrice d'incidence d'un graphe* est une matrice à $|V|$ lignes et $|E|$ colonnes pour laquelle le coefficient (v, e) est égal à 1 si le sommet v est une extrémité de l'arête e et vaut O sinon.

Question 3. Écrire le programme linéaire sous forme matricielle. (i.e. en faisant apparaitre vecteur de contraintes, vecteur objectif et matrice de contrainte).

Problème dual: Vertex Cover

Soit $G = (V, E)$ un graphe. Une *couverture par les sommets* (ou vertex cover) de G est un ensemble de sommets X tels que, pour toute arête (u, v), u ou v sont dans X (les deux peuvent l'être !). On essaie -bien sûr- de minimiser la taille du vertex cover en général.

Imaginons que l'on veuille surveiller le bon fonctionnement des arêtes d'un réseau (e.g. dans un réseau de capteurs, ou de robots). On désire placer des capteurs sur les sommets de façon à surveiller *toutes* les arêtes du graphe. Le tout en minimisant le coût total...

Question 4. Exprimer le problème du vertex cover minimal comme un problème de PLNE.

Question 5. Montrer que les contraintes "les variables sont dans $\{0, 1\}$" peuvent être remplacées par "les variables sont dans $\mathbb{N}$" sans modifier la solution optimale dans le PLNE de la Qestion 4. Similairement, pour le problème du couplage maximum, montrer que les contraintes "les variables sont dans $\{0, 1\}$" peuvent être remplacées par "les variables sont dans $\mathbb{N}$" sans modifier les solutions dans le PLNE de la Question 3.

Question 6. La relaxation fractionnaire d'un PLNE est le même programme linéaire sauf que les contraintes d'intégralité des variables on été supprimées (i.e. $x \in \mathbb{N}$ est remplacé par $x \in \mathbb{R}^+$).
Montrer que le dual de la relaxation fractionnaire de vertex cover est la relaxation du programme linéaire du couplage maximal.

Question 7. Donner une solution optimale du PLNE du vertex cover et couplage maximum dans le cas d'une clique à n sommets. Qu'en déduire sur le théorème de dualité forte dans le cas des PLNE?

Matrice d'incidence de biparti

La *matrice d'incidence d'un graphe* est une matrice à $|V|$ lignes et $|E|$ colonnes pour laquelle le coefficient (v, e) est égal à 1 si le sommet v est une extrémité de l'arête e et vaut O sinon.

Question 8.a. On a défini en cours une matrice d'incidence de biparti comme "une matrice avec deux 1 par ligne où on peut partitionner l'ensemble de colonnes en deux parties A et B de telle façon que l'un des deux 1 soit dans la partie de A et l'autre dans la partie de B". Montrer que la matrice d'incidence d'un graphe biparti est une matrice d'incidence de biparti.

Question 8.b. (*) Montrer que la matrice d'incidence d'un graphe est une matrice d'incidence de biparti si et seulement si le graphe est biparti.

Question 9. Qu'en déduire sur la complexité du problème VERTEX COVER MINIMUM et COUPLAGE MAXIMUM dans le cas des graphes bipartis? Et sur leurs valeurs respectives?

2 Implémentation

Matrice d'adjacence et d'incidence

La *matrice d'adjacence* A d'un graphe est une matrice à $|V|$ lignes et $|V|$ colonnes telle que si (i, j) est une arête alors $A(i, j) = A(j, i) = 1$. Sinon $A(i, j) = 0$.

En général les matrices d'adjacence de graphes sont creuses (le nombre d'arêtes dans les "vrais" graphes est souvent linéaire). Une représentation par liste plutôt qu'une représentation matricielle permet donc de gagner de l'espace.

Question 10.a. Écrire une fonction qui étant donné une matrice d'adjacence renvoie la matrice d'incidence.
Que renvoie votre programme pour un pour un cycle à 4 sommets, un chemin à 4 sommets, une clique à 4 sommets.

Question 10.b. Écrire une fonction qui étant donné la matrice d'incidence d'un graphe renvoie sa matrice d'adjacence.
Que renvoie votre programme pour un pour un cycle à 4 sommets, un chemin à 4 sommets, une clique à 4 sommets.

Attention: Chaque arête apparaît deux fois dans les matrices d'incidence: l'arête (i, j) apparaît en effet dans la ligne de la matrice d'adjacence de i et dans celle de j. Et cette arête ne doit apparaître qu'une seule fois dans la matrice d'incidence.

Implémentation - Programmation linéaire

La fonction LINPROG de Python permet de resoudre les programmes linéaires (bibliothèque scipy.optimize). Pour plus d'information sur linprog: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.linprog.html`.

Question 11.a. Écrire une fonction qui renvoie un couplage maximal **fractionnaire** étant donné un graphe G en entrée ainsi que sa valeur.

Que renvoie votre algorithme pour un cycle à 4 sommets, un chemin à 4 sommets, une clique à 4 sommets.

Question 11.b. Qu'est ce qui est renvoyé pour la relaxation fractionnaire du problème de couplage maximum? Vérifiez la cohérence de vos résultats.

Question 12. Quelles sont les solutions du problème pour la relaxation fractionnaire de "vertex cover minimal" et "couplage maximal" pour un cycle à 5 sommets. Comparer avec les solutions optimales entières.

Question 13. Un *graphe aléatoire d'Erdős-Renyi* $\mathcal{G}(n, p)$ à n sommets avec probabilité p sur les arêtes est un graphe à n sommets où chaque arête existe avec une probabilité p. Notez qu'on dit qu'il s'agit d'un graphe par abus de langage. Il s'agit en fait d'une distribution de probabilité sur l'ensemble des graphes.

Question 13.a. Écrire une fonction qui prend comme paramètre n et p et renvoie un graphe de $\mathcal{G}(n, p)$. Autrement dit, implémentez une fonction qui génère un graphe de taille n où l'existence de chaque arête est tirée avec probabilité p.

Question 13.b. Vérifiez la cohérence de vos résultats en implémentant une fonction qui compte le nombre d'arête dans le graphe.

Question 13.c. Donnez la valeur moyenne de la solution optimale de la relaxation fractionnaire couplage maximal (sur 10 tirages) pour $\mathcal{G}(30, 1/30)$, $\mathcal{G}(30, 1/10)$ et $\mathcal{G}(30, 1/2)$.
Que dire de la valeur moyenne de la solution optimale de la relaxation fractionnaire du vertex cover minimal?

Programmation linéaire en nombre entiers (**)

Question 14.a. (**) Une des méthodes pour résoudre un PLNE (Programme Linéaire en Nombre Entiers) consiste à résoudre optimalement sa relaxation fractionnaire. Si la solution optimale de la relaxation fractionnaire est entière, on a terminé. Sinon, il existe (au moins) une variable x dont la valeur optimale c est non entière. Alors on considère deux nouveaux Programmes Linéaires, un où on rajoute la contrainte $x \leq \lfloor c \rfloor$ et l'autre où l'on rajoute la contrainte $x \geq \lceil c \rceil$. On résout alors les deux PL et on renvoie la meilleur solution possible.

Implémenter l'algorithme ci-dessus qui, étant donné un PLNE, renvoie une solution optimale en nombre entier. Faites en sorte que votre algorithme renvoie aussi le nombre de PL (en nombre réel) vous avez du résoudre avant de trouver la solution optimale.

Question 14.b. (*) Une des méthodes pour améliorer l'efficacité de cet algorithme consiste à se souvenir de la valeur de la meilleure solution entière obtenue jusqu'à là. Si la solution fractionnaire obtenue à une certaine étape est moins bonne que la meilleure solution entière courante, on peut "couper la branche", i.e. aucune solution entière de cette branche ne sera meilleure.

Justifier la validité de cet algorithme et implémenter le. Dites combien de PL vous avez du résoudre avant de trouver la solution optimale pour une clique à 3, 5, 8 et 10 sommets pour le problème de couplage minimum. Comparez avec la question précédentes.

Question 13.c. (***) Avez-vous d'autres idées pour améliorer encore cet algorithme? Si oui, proposez votre méthode et implémentez là. Les résultats sont-ils convaincants?

Question 15.
Ecrire une fonction qui, étant donné un graphe (sous forme de listes d'adjacence), renvoie la valeur maximum d'un couplage.
Ecrire une fonction qui, étant donné un graphe (sous forme de listes d'adjacence), renvoie la valeur minimum d'un vertex cover.

Comparer les valeurs pour les cliques jusqu'à 10 sommets.

Question 15.a. (*) Conjecturez (et prouvez) la valeur d'un couplage maximum et d'un vertex cover minimum dans un graphe quelconque.

Question 15.b. (**) Prouvez que le ratio entre un couplage maximum et un vertex cover minimum est au plus 2.

Question 15.c. (**) Prouvez que l'écart ne peut pas être égal à 2 si le nombre de sommets est pair.