R5.11 – Méthodes d'optimisation pour l'aide à la décision

Guilherme Dias da Fonseca

TP 3

1 Sujet

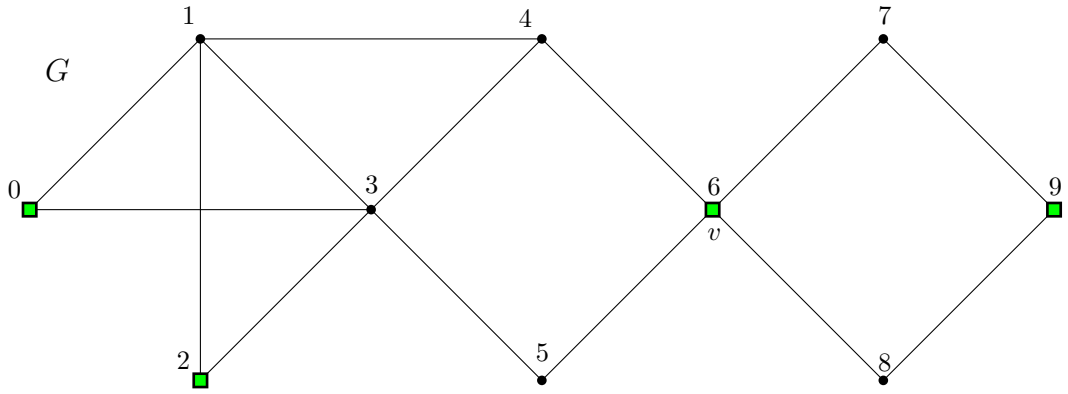
Ce TP continue le travail pour trouver un gros stable d'un graphe G. Durant le TP1, vous avez utilisé des algorithmes gloutons pour trouver des solutions faisables même pour des instances assez larges, mais pas toujours de très bonne qualité (stable pas suffisamment grand). Durant le TP3, vous avez utilisé la programmation linéaire en nombres entiers pour trouver des solutions optimales pour des instance de jusqu'à 20000 somment.

Ce TP sert à relier les deux premiers TPs, de façon à trouver de bonnes solutions pour les deux instances les plus larges. Voici l'idée. D'abord on trouve une solution faisable (TP1 ou TP2). Ensuite, on coupe un sous-graphe G' du graphe G. La taille t de G' doit être suffisamment petite pour que la programmation linéaire en nombres entiers arrive à bien résoudre le problème pour G', mais pas trop petite. (Une idée est d'initialiser t avec une petite valeur et augmenter t à chaque exécution de la boucle.) Finalement, on prend nouvelle la solution pour G' et on la colle dans G. On répète cette procédure plusieurs fois jusqu'à la limite de temps.

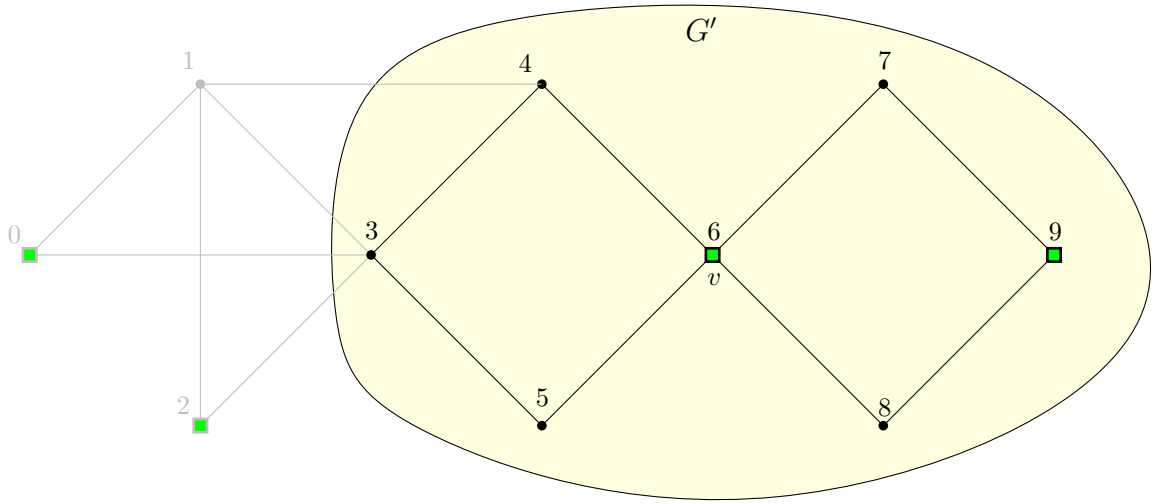
La méthode que vous allez utiliser pour découper le sous-graphe G' est très importante pour avoir une bonne solution. Vous pouvez par exemple choisir un sommet v au hasard et utiliser la procédure BFS de la classe graphe pour renvoyer les t sommets le plus près de v. Il faut penser qu'aucun sommet de G' doit avoir un voisin à l'extérieur de G' qui fait parti du stable, pour éviter d'avoir deux sommets adjacents dans la solutions. Pour éviter ce problème, ça suffit de supprimer de G' les sommets qui ont un voisin dans le stable à l'extérieur de G'.

1.1 Exemple

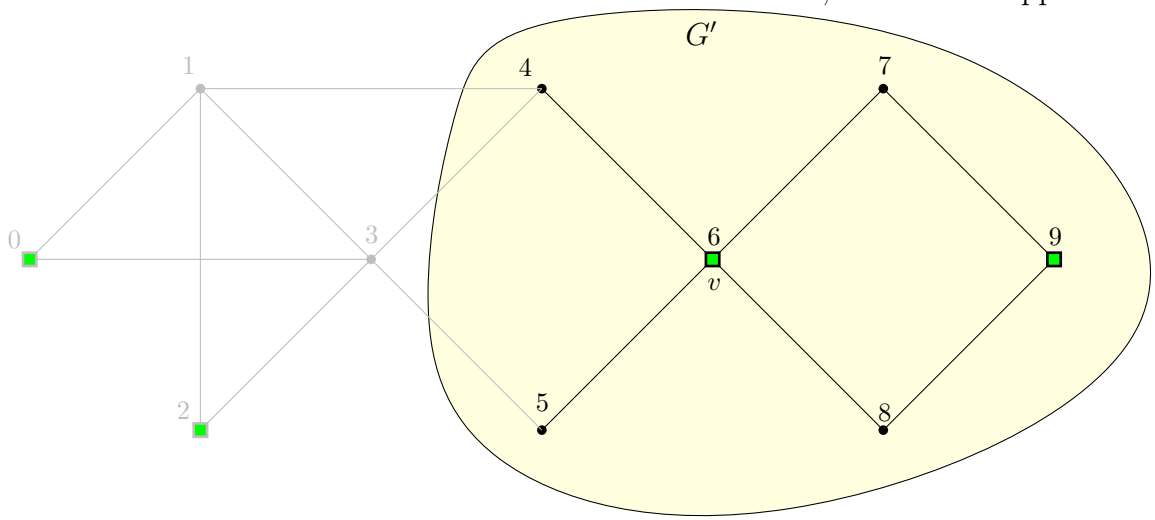
J'illustre l'idée avec un petit exemple. On commence avec un graphe G et un stable $I = \{0, 2, 6, 9\}$.



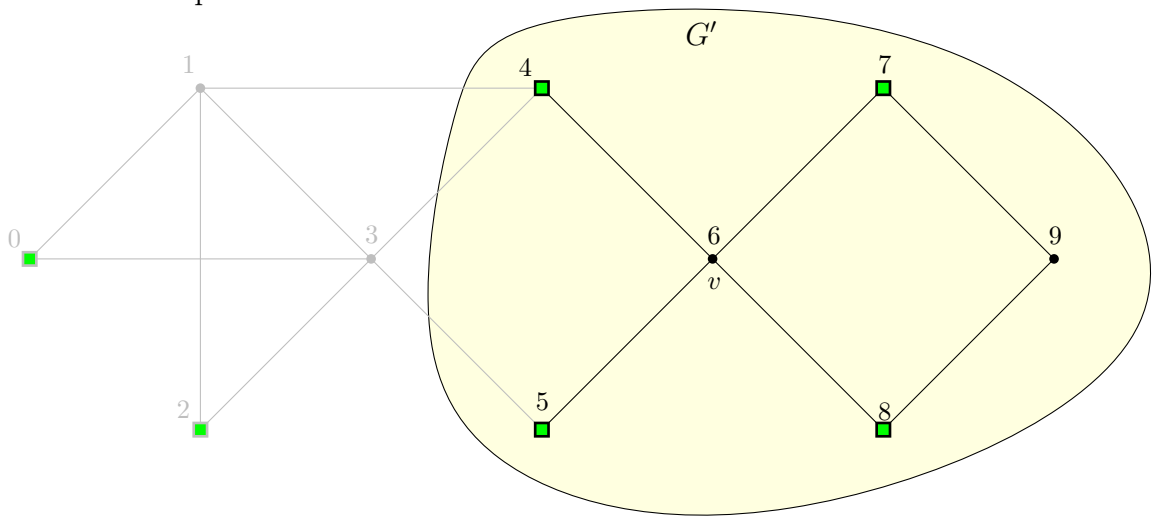
Ensuite, on choisi le sommet 6 et prend un sous-graphe G' à 7 sommets.



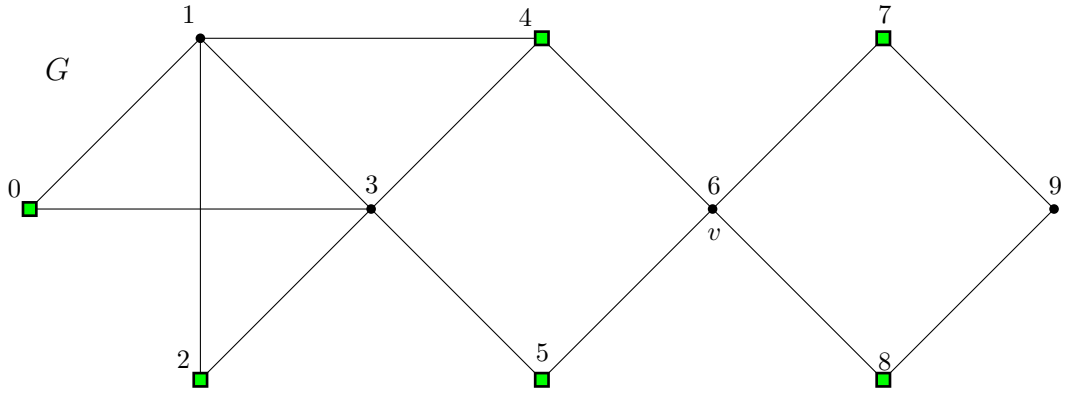
Le sommet 3 a un voisin 2 dans le stable à l'extérieur de G', alors on le supprime.



Avec la programmation linéaire en nombre entiers, on résout le problème pour G', qui trouve un stable de taille 4 pour G'.



On colle cette solution dans le graphe G et obtient $I = \{0, 2, 4, 5, 7, 8\}$.



1.2 Rendre le TP

Les fichiers d'entrée pour ce TP sont les fichiers avec 30000 et 40000 sommets. Votre code doit trouver une solution de base et la raffiner plusieurs fois. Vous devez sauvegarder seulement la meilleur solution trouvée. Vous avez une limite de temps de 5 minutes par fichier. C'est a vous de choisir comment obtenir la solution initiale, comment découper le graphe, quel taille de sous-graphe utiliser, comment limiter le temps de chaque appel du solver CPLEX etc.

Vous devez rendre un fichier zip avec :

1. Le code source avec votre nom dans la première ligne
2. Un script bash appelé `build.sh` pour compiler votre code sur Linux et exécuter votre logiciel avec tous les fichiers d'entrée en produisant automatiquement les fichiers de sortie.
3. Les fichiers de sortie avec le même nom des fichiers d'entrée correspondants mais l'extension `.ind` à la place de `.edges`.

Pensez que j'ai beaucoup de TPs à corriger, alors pour que je puisse rendre cette tache plus efficace je vais coder des logiciels pour faire quelques parties automatiquement. Pour ça, il faut avoir exactement les bons noms des fichiers (minuscule et majuscule c'est pas pareil). Parfois, je vois des trucs très bizarres comme des fichiers rar renommé zip pour qu'Ametice l'accepte et bien sur que le script que j'écris pour décompresser les fichiers ne marche pas ! Le non respect des consignes peut être pénalisé.