Operations Research Project

Matthias Carré

Mai 2025

Contents

1 Introduction

Ce projet a pour but de travailler avec les graphes et d'implémenter des algorithmes sur ces derniers. Par simplicité, le choix du langage s'est porté sur Python, qui permet de travailler facilement avec des structures de données variées.

2 Structure

2.1 Parser

La première partie du programme est le parseur. Il s'occupe de lire le fichier texte contenant le graphe et de le transformer en structures Python sous la forme :

- Un dictionnaire contenant :
  - le nombre de nœuds,
  - le nombre d'arcs,
  - le nœud source,
  - le nœud puits.
- Un tableau contenant, pour chaque arc :
  - le nœud de départ,
  - le nœud d'arrivée,
  - le flow maximum,
  - le coût de l'arc.

2.2 Scripts

La structure ainsi créée est ensuite utilisée par les autres programmes. Le tableau est transformé en liste d'adjacence sous la forme : un dictionnaire Python contenant, pour chaque nœud, un tableau avec :

- le sommet d'arrivée,
- le flow maximum,
- le coût de l'arc,
- le flow utilisé.

3 Implémentations

3.1 Fonctions Utiles Implémentées

3.1.1 DFS

Deux fonctions `dfs` sont implémentées :

- **dfs** : renvoie la liste d'arcs formant un chemin de la source vers le puits, en s'arrêtant au premier trouvé.
- **dfsAccessible** : renvoie la liste des sommets accessibles depuis un sommet donné.

3.1.2 Fonctions sur les Arcs

- **getEdge** : renvoie les informations de l'arc entre deux sommets donnés.
- **reduceFlowEdge** : réduit le flow de l'arc entre deux sommets donnés de la valeur donnée. Si un arc est saturé, il est retiré et ajouté à une liste.
- **increaseFlowEdge** : augmente le flow de l'arc entre deux sommets donnés de la valeur donnée.

3.1.3 Résiduel

Prend une liste d'adjacence et un chemin pour créer le graphe résiduel en réduisant les capacités des arcs du chemin donné.

3.1.4 Dijkstra

Prend une liste d'adjacence et une source et renvoie la distance de tous les nœuds depuis la source.

3.2 Flow Maximum

Pour calculer le flow maximum du graphe, les étapes à répéter sont :

- Chercher un chemin de la source vers le puits grâce à un `dfs`.
- Si un chemin est trouvé, diminuer la capacité de tous les arcs du chemin par le minimum des capacities sur ce chemin.
- Calculer le graphe résiduel et répéter l'action.

Lorsque plus aucun chemin n'est trouvable, le graphe est saturé. Il suffit alors d'additionner les valeurs des flows sortants de la source pour obtenir la valeur du flow maximum.

En pratique dans le code, on travaille avec deux graphes :

- le premier où l'on met à jour uniquement les valeurs de flows des arcs,
- et le résiduel où l'on retire les arcs saturés et sur lequel on effectue les `dfs`.

Cela permet, une fois qu'il n'y a plus de chemin disponible, d'utiliser le premier graphe pour obtenir les informations sur les capacités restantes.

3.3 Coupe Minimale

La coupe minimale est liée au flow maximum ; c'est pourquoi les deux sont calculés dans le même programme. On garde les mêmes étapes, mais en plus, on sauvegarde les arcs retirés du graphe de base.

À la fin de l'algorithme, on regarde quels arcs retirés possèdent un sommet accessible depuis la source et l'autre non accessible. Pour cela, on utilise un `dfs`. Ainsi, on déduit la liste des arcs formant la coupe minimale du graphe.

3.4 Flow Maximum de Coût Minimum

Pour calculer le flow maximum de coût minimum, on répète les étapes suivantes :

- Rechercher le chemin de coût minimum de la source jusqu'au puits grâce à Dijkstra.
- S'il existe, calculer le résiduel du graphe en réduisant les capacités selon ce chemin.
- Modifier les coûts des arcs avec la formule :

$$c_{i,j} \leftarrow d(i) + c_{i,j} - d(j)$$

  où d est la distance depuis la source et c le coût de l'arc.

Une fois qu'aucun chemin n'est trouvable, on regarde la somme des flows sortants de la source pour obtenir le flow maximum. Enfin, on calcule le coût final via :

$$\sum_{e \in G} c(e) \times f(e)$$

avec $c(e)$ le coût de l'arc e et $f(e)$ le flow passant par e.

4 Conclusion

Ce projet a permis de mieux comprendre les algorithmes liés aux graphes et leur fonctionnement. L'utilisation de Python a été le choix le plus simple et logique en raison de sa simplicité dans la gestion des données, même s'il reste moins performant lorsque l'on travaille avec des ensembles de données de grande taille.