UFR Mathématiques et Informatique

Projet — Flot de coût minimum

Graphes et Algorithmes
Master ROAD
2022–2023

Le projet est à rendre pour le 9 Janvier 2023
Vous déposerez vos fichiers source `.cpp` (et `.h` éventuels), ainsi qu'un rapport `.pdf`. **Commentez** votre code.

L'objectif du projet est de calculer le flot de coût minimum dans un graphe. Le résultat sera **le flot circulant sur chaque arc du graphe**, ainsi que le **coût total du flot**. Il y a deux algorithmes proposés dans le cours pour résoudre ce problème :

- l'algorithme d'**élimination de circuits** (transparent 15 du cours *Problème de flot de coût minimum et premier algorithme de résolution*) ;
- l'algorithme **des plus courts chemins successifs** (transparent 14 du cours *Algorithmes de résolution du problème de flot de coût minimum basés sur les coûts réduits*).

Ce projet contiendra trois parties :

1. la lecture des fichiers d'instance, selon le format décrit ci-dessous,
2. la solution du problème de flot de coût minimum, en utilisant **l'un des deux algorithmes au choix**,
3. la vérification de la correction de la solution.

Il faudra être attentif à la structure de données utilisée pour représenter le graphe. Comme indiqué dans le cours, le graphe résiduel peut être un **multigraphe**, avec plusieurs arcs entre le même nœud source et le même nœud destination.

1 Format du fichier d'entrée

Les fichiers d'entrée sont des fichiers textes, dans lesquels chaque ligne commence par une lettre parmi `c`, `p`, `n`, ou `a` pour indiquer quelle information est contenue dans cette ligne.

Les lignes sont formattées comme suit :

`c` indique une ligne de commentaire. Les fichiers contiennent souvent un en-tête de commentaires, qui décrit les paramètres utilisés pour générer cette instance.

`p` indique la définition du problème. Le format est : `p min NODES DENSITY`, où :
- `min` indique que c'est un problème de flot de coût minimum ;
- `NODES` est le nombre total de nœuds ;
- `DENSITY` est le nombre total d'arcs.

`n` indique une définition de nœud. Les définitions de nœuds sont données après la définition du problème, avec le format `n ID SUPPLY`, où :
- `ID` est un index numérique unique du nœud, chaque nœud est numéroté en commençant à 1 ;
- `SUPPLY` est la valeur du flot produit au nœud (positive pour les nœuds producteurs, négatives pour les nœuds consommateurs). Pour gagner de la place, seuls les nœuds dont la valeur de flot produit est non-nulle sont présents dans le fichier.

`a` indique une définition d'arc. Les définition d'arc sont données après les définitions de nœud, et ont le format `a FROM TO MINCAP MAXCAP COST`, où :
- `FROM` et `TO` sont les indices des nœuds respectivement d'origine et de destination de l'arc ;
- `MINCAP` et `MAXCAP` sont les capacités respectivement minimales et maximales de l'arc ;
- `COST` est le coût unitaire de l'arc.

Dans les instances fournies pour ce projet, la capacité minimale `MINCAP` sera toujours 0, et le coût d'un arc `COST` sera toujours positif ou nul.

Il vous faudra décrire la structure de données utilisée pour stocker le graphe, et écrire une ou plusieurs fonctions pour lire un fichier d'entrée selon ce format et renvoyer le graphe correspondant.

2 Algorithme de résolution

Comme indiqué plus haut, vous pouvez choisir entre les deux algorithmes (élimination de circuits ou plus courts chemins successifs). Quelques points à noter :

— L'algorithme d'élimination de circuits requiert d'identifier un circuit de coût négatif, et suggère d'utiliser l'algorithme de Bellman-Ford. Cet algorithme détecte en effet la présence d'un circuit de coût négatif *accessible depuis la source*. Il vous faudra donc choisir une source arbitraire. Même s'il ne détecte aucun circuit de coût négatif, il est possible qu'il existe un tel circuit, mais non accessible depuis cette source. Décrivez (en commentaires) le procédé utilisé pour détecter un circuit de coût négatif en utilisant l'algorithme de Bellman-Ford.

— Dans l'algorithme des plus courts chemins successifs, il faut trouver un nœud k et un nœud l tels qu'il existe un chemin de k à l (ligne 5). Décrivez et justifiez de la procédure utilisée pour choisir ce nœud k de départ, ou d'un prétraitement appliqué au graphe pour éviter d'avoir à choisir ce noeud de départ.

3 Vérification que le résultat est correct

Il vous est également demandé d'implémenter une fonction de **validation** d'une solution. Il faudra que cette fonction vérifie que la solution respecte bien toutes les contraintes du problème de flot de coût minimum, comme indiqué au transparent 5 du cours *Problème de flot de coût minimum et premier algorithme de résolution*. Cette fonction calculera également le **coût** de la solution retournée.

4 Fichiers d'instance

Un ensemble de fichiers d'instance sera disponible sur Moodle, avec les valeurs du flot de coût minimum pour chacune.

5 Rapport

En plus du code de votre programme, vous devrez rendre un petit document écrit qui

— explique très brièvement ce qu'il y a dans chaque fichier ;

— décrit les bugs non corrigés et les difficultés rencontrées ;

— indique les instances sur lesquelles vous avez testé votre algorithme et obtenu le même résultat que celui dans le fichier fourni.