DUT Informatique Graphique
Le Puy en Velay

Graphe
TP 4

département informatique graphique

Imagerie 1 — Pierre-Yves Bischoff — 2021

Algorithme de Ford - Fulkerson

Ce TP va vous permettre de résoudre un problème d'optimisation très régulièrement utilisé : comment faire passer le maximum (de voitures, de fret, de piétons, ...) dans un réseau. Ici le réseau sera représenté par un graphe dont les arêtes vont représenter le flux maximal pouvant passer entre 2 nœuds.
Dans le problème le graphe sera orienté. (Les arêtes iront du FisrtNode au SecondNode). Nous rajouterons une donnée membre sur l'arête qui contiendra le flux actuel qui la traverse.
Dans ce TP il ne pourra y avoir **qu'une seule arête entre 2 noeuds** et pas de boucle.
Le nœud de départ n'a pas d'arêtes incidentes et le nœud d'arrivée d'arêtes sortantes.

Exercice 1 : CEdge

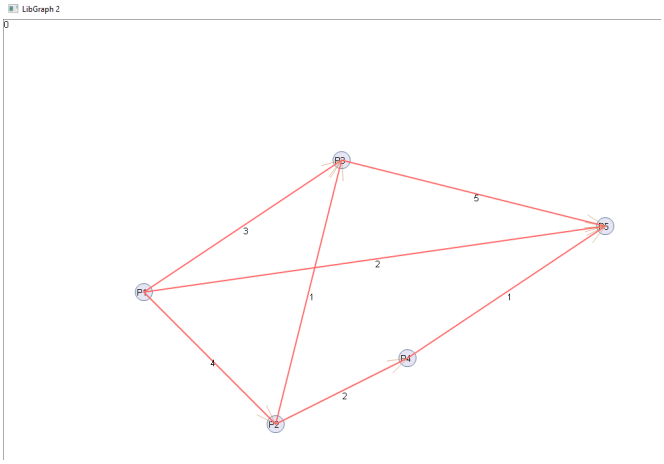
J'ai épuré la classe CEdge en modifiant le type de donnée de `m_weight` en `size_t`.
Je vous rappelle juste que le graphe est orienté (le nœud de départ est le FirstNode et le nœud d'arrivée le SecondNode).
On ajoute une variable `m_flux` qui contiendra le flux actuel qui passe dans l'arête (0 par défaut et inférieur ou égal au poids ! ! ! !). En effet le poids de l'arête représente le flux maximal autorisé.
Vous compléterez les accesseurs sur cette donnée : `GetFlux, AddFlux, SubFlux`.

Exercice 2 : prog.cpp

C'est le fichier de la fonction *main*. Il gère tous les événements. Le seul cas que vous aurez à programmer c'est le lancement de l'algorithme de Ford-Fulkerson en appuyant sur 'F' .
Vous devrez gérer la sélection du nœud de départ et du nœud d'arrivée.
Pour ce faire vous devrez peut-être rajouter une étape dans l'énumération `Etape`.

Exercice 3 : CTP

La classe CTP va faire la liaison entre le graphe qu'elle possède en donnée membre et les fonctions de feedback Elle va gérer toutes les fonctionnalités de base : Ajouter arête et nœud, et les supprimer. Elle permet aussi de dessiner le Graphe.
Mettez le nœud de départ en vert et le nœud d'arrivée en rouge dans le `DrawNode`.
Modifiez le `DrawEdge` pour afficher le flux passant par l'arête en plus du poids.
Modifiez le `Draw` pour afficher le `fluxMaximum` arrivant au nœud d'arrivée en haut à gauche.
Voici le graphe que vous devriez obtenir en lançant le programme.



Exercice 4 : CGraph

Vous avez à disposition une fonction qui crée le graphe d'exemple.
Cette classe est la plus compliquée je vous conseille de commencer par faire les fonctions faciles.
On doit rajouter une donnée pour l'orientation (bool) ainsi que les accesseurs sur cette donnée.
Vous devez ajouter deux fonctions `GetEdgesIn` et `GetEdgesOut` qui prennent un nœud en entrée et renvoie la liste des arêtes incidentes et sortantes.
Vous aurez à rajouter la **Fonction principale du TP :** `OptimisationFF` qui fera l'algorithme de Ford Fulkerson.
Je vous conseille de regarder cette vidéo qui explique l'algorithme en détail :
`https://www.youtube.com/watch?v=eL3fTl4mykY`
Description de l'algorithme sur cet exemple.

- On choisit P1 comme sommet de départ et P5 comme sommet d'arrivée.
- On met tous les flux courants des arêtes à la valeur 0
- On commence par la chaîne P1P2P4P5 et on a P1P2 4 P2P4 2 P4P5 1 donc le minimum c'est 1 et on rajoute 1 à la valeur de tous les flux de ces 3 arêtes.
- Puis on prend P1P2P3P5 et on a P1P2 3 (4-1) P2P3 1 P3P5 5 donc le minimum c'est 1 et on rajoute 1 à ces 3 arêtes
- Puis on prend P1P3P5 et on a P1P3 3 P3P5 4(5-1) donc le minimum c'est 3 et on rajoute 3 à ces 2 arêtes.
- Il reste P1P5 et on a P1P5 2 donc on rajoute 2 à cette arête.
- ...
- Il n'y a pas d'autre chaîne améliorante donc le flux maximal arrivant en P5 est 4 (arête P3P5) + 2 (arête P1P5) + 1 arête P4P5 soit 7 ! ! !

Pour programmer cet algorithme il faut trouver toutes les chaînes commençant par le nœud de départ et finissant au nœud d'arrivée.
Vous pouvez utiliser la fonction de graphe qui prend en entrée le nœud de départ et le nœud d'arrivée et qui renvoie une liste de chaîne.
Une chaîne est une liste de nœuds qu'on peut convertir en liste d'arêtes avec leur sens.
On parcourt la liste des chaines commençant par le nœud de départ et finissant par le nœud d'arrivée.
Pour chaque chaine on parcourt la liste des arêtes avec leur orientation et on cherche si c'est une chaine améliorante (le minimum de flux de chaque arête est > 0 (strictement).
Si elle améliore on ajoute ce fluxAmeliorant aux arêtes dans le bon sens (true) ou on enlève ce fluxAmeliorant aux arêtes dans le mauvais sens (false).