

Recherche de chemin de valeur minimale --- Algorithme de Bellman

Travail à effectuer

Vous devez mettre en œuvre un programme qui :

- lit les arcs d'un graphe dans un fichier texte et en sauvegarde le contenu dans une structure de données de votre choix ;
- imprime le contenu de cette structure de donnée ;
- exécute l'algorithme de Bellman pour calculer les chemins de plus faible valeur partant d'un sommet indiqué par l'utilisateur du programme ;
- affiche les valeurs calculées (longueur du chemin et prédécesseur du sommet) à chaque étape du déroulement de l'algorithme ;
- affiche le cas échéant la présence de circuit absorbant.

Graphes

Il s'agit de graphes orientés et valués.

Les sommets sont représentés par les nombres entiers de 0 à N-1 pour un graphe comportant N sommets.

Les valeurs associées aux arcs sont des nombres entiers quelconques (donc valeurs nulles ou négatives possibles).

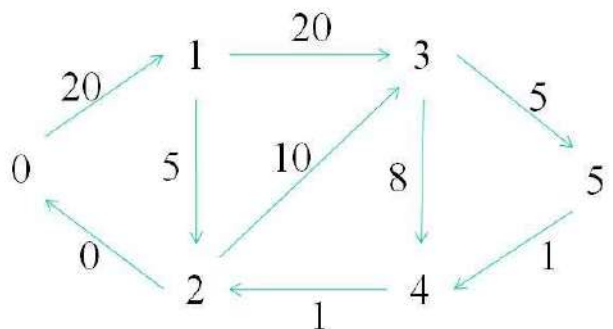
Fichier de données

Nous vous proposons la structure suivante, mais vous pouvez bien entendu l'adapter à vos besoins, ou faire quelque chose de totalement différent !

Ligne 1	nombre de sommets
Lignes 2 et suivantes	arcs sous la forme extrémité_initiale extrémité_terminale valeur
Dernière ligne	-1

Pour le graphe G24 donné en TD, le fichier est le suivant :

```
6
0 1 20
1 3 20
1 2 5
2 0 0
2 3 10
3 5 5
3 4 8
4 2 1
5 4 1
-1
```



Nous vous donnons en annexe le code d'une fonction pouvant servir de base à votre procédure de sauvegarde dans votre structure de donnée.

Rappel – Méthode de Bellman

Soit le calcul des chemins de plus faible valeur en partant du sommet D.

On initialise les longueurs de la façon suivante :

$$\pi_0(D) \leftarrow 0$$

$$\pi_0(x) \leftarrow +\infty \quad \text{pour tout sommet } x \text{ différent de } D$$

On initialise les prédécesseurs de chaque sommet de la façon suivante :

$$\text{Pred}(D) \leftarrow D$$

$$\text{Pred}(x) \leftarrow \text{indéfini} \quad \text{pour tout sommet } x \text{ différent de } D$$

A chaque itération k, on effectue les modifications suivantes :

Pour tout sommet x :

$$\pi_k(x) \leftarrow \min [\pi_{k-1}(x), \min_{y \in \Gamma^{-1}(x)} [\pi_{k-1}(y) + v(y,x)]]$$

où $v(y,x)$ est la valeur associée à l'arc (y,x)

si $\pi_k(x) \neq \pi_{k-1}(x)$, alors $\text{Pred}(x)$ devient le prédécesseur y tel que $\pi_k(x) = \pi_{k-1}(y) + v(y,x)$

Conditions d'arrêt :

Valeurs π correctes au rang k si pour tout sommet on a $\pi_k(x) = \pi_{k-1}(x)$

Abandon (présence de circuit absorbant) si on atteint une itération $k > \text{nombre de sommets du graphe}$.

Langage de programmation

C

C++ est autorisé (par exemple si vous utilisez un environnement de développement « C++ ») mais pas l'utilisation des classes d'objets, ni des types 'list', 'map', ...

Bien évidemment, aucune librairie de mise en œuvre et manipulation de graphe trouvée sur internet !

Rendu du travail

Vous devrez exécuter votre programme sur les graphes qui vous seront donnée en séance de TP.

Vous devrez adresser, à une adresse qui vous sera communiquée lors de la séance de TP, un email avec les caractéristiques suivantes :

Sujet : EFREI L3 TG TP1 – Nom1, Nom2

Message : traces d'exécution en clair

Texte uniquement, en utilisant une police à taille fixe (par exemple Courier New).

Pour copier le contenu d'une fenêtre « DOS » d'exécution de votre programme, vous devez effectuer les opérations suivantes :

- Curseur de la souris dans la fenêtre
- Clic droit, menu « sélectionner tout »
- Entrée

Par de copie d'écran en mode image.

Pièces jointes : fichiers de code source + fichiers de données si vous en avez changé la structure

Code source uniquement, pas d'exécutable.

Pas de fichier compressé archive : mettez plusieurs pièces jointes si nécessaire.

Tous vos fichiers doivent être préfixés par vos deux noms, y compris les fichiers de données si vous en avez changé la structure et que vous nous les communiquez.

Date limite d'envoi : 72 heures (samedi et dimanche non compris) après la fin de votre séance de TP.

Tout manquement à ces consignes sera pénalisé !

Exemple d'email :

Sujet : EFREI L3 TG TP1 – Barbot, Lepoivre

Pièces jointes : barbot-lepoivre-tp1-bellman.cpp

Message :

G24.txt

Matrice d'adjacence

ligne x / colonne y =

valeur de l'arc (x,y) s'il existe

vide sinon

	0	1	2	3	4	5
0		20				

1				5	20			
2	0				10			
3						8	5	
4				1				
5						1		

BELLMAN (1)

ligne k / colonne x =

format « valeur predecesseur »

vide si valeur = infini

k\	0	1	2	3	4	5
0		0	1			
1		0	1	5	1	20
2	5	2	0	1	5	1
3	5	2	0	1	5	1
4	5	2	0	1	5	1
5	5	2	0	1	5	1

Annexe – Fonction de lecture de fichier et sauvegarde dans une structure de données

Compte-tenu de la structure de fichier donnée précédemment, le code suivant vous propose une trame de procédure de lecture. A vous de l'adapter à votre programme.

```
void lire_graphe ( ...ajoutez les paramètres qui vous semblent utiles... )
{
    ifstream source_graphe ( "G24.txt" ) ;
    // Nombre de sommets du graphe
    int nb_sommets
    // Arcs du graphe
    int extremite_initiale ;
    int extremite_terminale ;
    int valeur_arc ;
    // Lecture nombre de sommets
    source_graphe >> nb_sommets ;
    ... ajoutez ici votre code pour stocker cette valeur...
    // Lecture des arcs
    if ( nb_sommets > 0 ) {
        source_graphe >> extremite_initiale ;
        while ( extremite_initiale != -1 ) {
            source_graphe >> extremite_terminale ;
            source_graphe >> valeur_arc ;
            if ( ( extremite_initiale >= 0 )
                && ( extremite_initiale < nb_sommets )
                && ( extremite_terminale >= 0 )
                && ( extremite_terminale < nb_sommets ) )
            {
                ... ajoutez ici votre code pour stocker cet arc...
            } ;
            source_graphe >> extremite_initiale ;
        } ;
    } ;
} ;
```