EFREI 2018/2019
L3 – L3 Afrique – L3 Chine

Théorie des Graphes
Projet

Objectif principal : Recherche de chemin de valeur minimale

Description du travail à effectuer

Graphes à prendre en compte

Votre programme doit être capable de traiter correctement tout graphe correspondant aux critères suivants :

- Le graphe est orienté
- Les sommets sont représentés par des nombres à partir de '0'
  Il n'y a pas de rupture dans la numérotation des sommets (par exemple 15 sommets seront numérotés de 0 à 14).
- Le graphe est valué :
  1 valeur numérique par arc ; pas de restriction sur les valeurs possibles : elles peuvent être négatives, nulles ou positives
- Au plus 1 arc d'un sommet 'x' vers un sommet 'y'
- Peut contenir des boucles
- Peut contenir un ou plusieurs sommets sans prédécesseur, ainsi qu'un ou plusieurs sommets sans successeur

Programme principal

Votre programme doit permettre, sans le redémarrer, d'effectuer des tests sur plusieurs graphes.

début
    tant que l'utilisateur veut tester un nouveau graphe *faire*
        choisir le fichier texte par son numéro
        lire le fichier correspondant et sauvegarder le graphe en mémoire
        imprimer le graphe sous forme d'une matrice d'adjacence à partir de la mémoire et non pas lors de la lecture du fichier texte
        calculer les chemins de valeur minimale
    fait
fin

Lecture du fichier

Les graphes de test seront décrits dans des fichiers « .txt » que votre programme devra lire. Les fichiers doivent être nommés de façon suivante :

L3-<numéro d'équipe>-#.txt

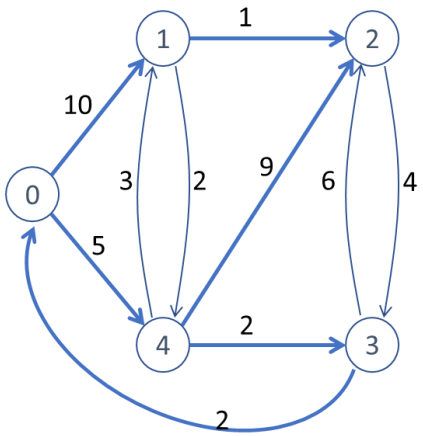
où le # sera remplacé par le numéro du graphe texte. Par exemple, si vous êtes l'équipe B10 et qu'il s'agisse du graphe de test numéro 8, le fichier doit être nommé L3-B10-8.txt.

Structure de fichier :

En première ligne le nombre de sommets
Suivie d'une ligne par arc, chaque ligne sous la forme :
extrémité_initiale valeur_de_l'arc extrémité_terminale

Exemple :

```
5
0 5 4
0 10 1
1 2 4
1 1 2
2 4 3
3 6 2
3 2 0
4 3 1
4 9 2
4 2 3
```



Si vous le jugez utile, vous pouvez ajouter le nombre d'arcs sur la première ligne, après le nombre de sommets.

Remarque : le nombre de sommets est impératif pour permettre l'existence de sommets isolés. Par exemple, le graphe ci-dessus doit pouvoir être modifié en indiquant '6' au lieu de '5', ce qui aurait pour effet d'ajouter un sixième sommet (de numéro '5' !) qui n'a ni prédécesseur, ni successeur.

Les informations doivent être mises dans la mémoire dans une structure de donnée de votre choix, et aucune relecture du fichier texte n'est plus permise.

Imprimer le graphe sous forme de matrice d'adjacence et matrice des valeurs

Le graphe lu doit être affiché, à partir de la mémoire, sur l'écran sous forme de deux matrices :

- une matrice d'adjacence booléenne, avec, préférablement, les 0 remplacés par du vide ou des traits sur l'écran,
- et une matrice des valeurs où là aussi les arcs non existants sont préférablement remplacés par du vide ou des traits.

Il vaut mieux que ces deux matrices soient organisées en colonnes bien définies (sans décalage dû à la longueur des nombres).

Doivent également être affichés le nombre de sommets et le nombre d'arcs.

Calcul des chemins de valeurs minimales

Votre programme commence par détecter la présence ou non d'arc à valeur négative.

En cas d'arc à valeur négative, il en informe l'utilisateur et il exécute l'algorithme de Bellman.

En absence d'arc à valeur négative, il demande à l'utilisateur quel algorithme utiliser.

Dans tous les cas, il demande à l'utilisateur de décider du sommet de départ des chemins recherchés.

```
si le graphe contient au moins un arc avec une valeur négative
alors    demander à l'utilisateur le sommet de départ
         exécuter l'algorithme de Bellman
sinon    demander à l'utilisateur le sommet de départ
         demander à l'utilisateur l'algorithme à utiliser (Dijkstra ou Bellman)
         exécuter l'algorithme choisi
finsi
```

Attention : Vous devez mettre en œuvre les 2 méthodes : Dijkstra et Bellman.

Lors de l'exécution de ces algorithmes, votre programme doit afficher les choix et résultats effectués à chaque étape (comme fait en cours et TD). Par exemple, pour l'algorithme de Dijkstra, on peut avoir l'affichage suivant (où, évidemment, les $+\infty$ seront remplacés par du vide, et les gros points, par un caractère bien lisible de votre choix) :

CC	0	1	2	3	4
0	•	10 (0)	∞	∞	5 (0)
0 4	•	8 (4)	14 (4)	7 (4)	•
0 3 4	•	8 (4)	13 (3)	•	•
0 1 3 4	•	•	9 (1)	•	•
0 1 2 3 4	•	•	•	•	•

Les chemins eux-mêmes doivent aussi figurer sur l'écran : par exemple, le chemin le plus court de 0 à 2, de longueur 9, est 012.

Si le graphe contient un circuit absorbant, cela doit être détecté par l'algorithme de Bellman qui doit en informer l'utilisateur et, évidemment, dans ce cas il ne doit pas afficher les chemins les plus courts.

Tout ce qui figure sur l'écran, doit en même temps être écrit dans un fichier L3-<numéro d'équipe>- trace#_#.txt, où le premier # doit être remplacé par le numéro du graphe test, et le second, par le numéro du sommet d'origine. Par exemple, si vous exécutez la recherche des chemins les plus courts sur le graphe 5 depuis le sommet 3, le fichier créé doit s'appeler L3-<numéro d'équipe>-trace5_3.txt.

Organisation du travail

Travail par équipes

Le nombre d'équipes par groupe TD/TP, le nombre d'étudiants par équipe, ainsi que la date limite de constitution des équipes vous seront communiqués par votre enseignant TD/TP.

Séance de TP

Une séance de TP est prévue pour vous permettre de progresser dans vos développements avec le soutien de votre enseignant.

Soutenances

Le planning des soutenances vous sera communiqué plus tard.

Des graphes de test vous seront fournis avant la soutenance sous forme graphique, dans un délai suffisant pour vous permettre de les coder par des fichiers texte. Ces fichiers devront impérativement être préparés à l'avance, et ils feront partie du rendu.

Rendu du travail

Toutes les équipes devront remettre leur travail à leur enseignant à la même date. Des consignes vous seront communiquées ultérieurement.

Le contenu du rendu :

- Tout fichier code que vous avez tapé vous-même (**donc aucun fichier produit par le logiciel durant la compilation ou exécution)**, bien commenté. Tout fichier que vous utilisez doit être nommé de façon à contenir le même préfixe L3-<numéro d'équipe> : par exemple, si vous avez un fichier « main » et si vous utiliser le langage C++, ce fichier doit avoir le nom L3-<numéro d'équipe>-main.cpp.
- Tous les fichiers .txt des graphes de test (oui, malgré le fait que ce sont vos enseignants qui vous ont fourni les graphes de test), dans le même répertoire que le code.
- Un ppt ou pdf de la présentation,
- Quelques traces d'exécution (au moins un par graphe de test).

Langage de programmation

Au choix : C, C++, Python, Java