Rapport Projet d’Informatique :

Graphe, Plus court chemin et métro

CELIE Caroline – GUILLAUME Laure

Introduction :

Ce projet consiste en la réalisation d’un programme permettant de déterminer le plus court chemin entre deux stations de métro parisien. Ce programme se basera sur une structure de graphe (représentée ci-dessous). Chaque station représentera un nœud de ce graphe et chaque trajet possible entre deux stations consecutives sera représenté par les arcs de ce graphe.

Il nous faut donc réaliser pour ce projet cette structure de graphe accompagnée des fonctions la rendant utilisable, ainsi que la fonction de lecture d’un ifhcier regroupant les sattions de métro et de la fonction représentant l’algorithme de plus court chemin de Belleman.

Structures utilisées

La structure la plus utilisée est celle des Graphes :

*typedef struct {*

*unsigned char nom\_station[50];*

*unsigned char nom\_ligne;*

*unsigned int num\_station;*

*double poids\_noeud;*

*Liste arc;*

*}\* sommet;*

*#endif*

*#ifndef \_Graphe*

*#define \_Graphe*

*typedef struct {*

*sommet stations;*

*unsigned int nX;*

*unsigned int nA;*

*}\* Graphe;*

Cette structure est composée de deux entiers non signés nX et nA qui représente respectivement le nombre de sommetss du graphe et le nombre total d’arc du graphe, ainsi que d’un pointeur sur un tableau qui est représenté ici par la structure sommet. Ce tableau donne pour chacune de ses lignes, le numéro, le nom du sommet, la ligne à laquelle il appartient, le poids du sommet ainsi que les arcs partant de ce sommet.

Les arcs sont représentés par une liste d’éléments comprenant le sommet d’arrivée et le poids de l’arc en question.

La structure élément est décrite ci-dessous :

*#ifndef \_ELEMENT*

*#define \_ELEMENT*

*typedef struct{*

*unsigned int Xdest; //Numero du sommet d’arrivée*

*double poids\_arc; //poids de l'arc ainsi realise*

*} ELEMENT;*

*#endif // \_ELEMENT*

Fonctions et explications

Différentes fonctions vont devoir être réalisé durant ce projet :

-Les fonctions liées à la structure de Graphe. En effet, ayant défini une nouvelle structure, nous devons définir les fonctions la caractérisant et nous permettant de remplir ce graphe. Nous avons donc les fonctions suivantes :

*Graphe nouveau\_graphe(unsigned int nX,unsigned int nA);*

Fonction créant un graphe possédant nX sommets et nA arcs.

*void affiche\_graphe(Graphe g);*

Fonction d’affichage du graphe

*void detruit\_graphe(Graphe g);*Fonction de destruction de graphe

*void graphe\_ecrit\_nX(Graphe g, unsigned int nX);*Fonction permettant de changer le nombre de sommets du graphe

*void graphe\_ecrit\_nA(Graphe g, unsigned int nA);*

Fonction permettant de modifier le nombre total d’arc du graphe

*unsigned int graphe\_lit\_nX(Graphe g);*

Fonction permettant de lire le nombre de sommets du graphe

*unsigned int graphe\_lit\_nA(Graphe g);*

Fonction permettant de lire le nombre total d’arc du graphe

*double graphe\_lit\_poids(Graphe g, unsigned int u);*

Focntion permettant de lire le poids du sommet u

*void graphe\_ecrit\_poids(Graphe g, unsigned int u, double val);*

*Fonction permettant d’écrire le poids du sommet u et d’y mettre le réel val*

*void graphe\_ecrit\_poids\_arc(Graphe g, unsigned int u, unsigned int v, double val);*

Fonction permettant d’écrire le poids de l’arc ayant pour sommet de déaprt u et pour sommet de destination v et d’y joinder la valeur val.

*double graphe\_lit\_poids\_arc(Graphe g, unsigned int u, unsigned int v);*

*foncrion permettant de lire le poids de l’arc partant du somemt u et arrivant au sommet v*

*void graphe\_ajoute\_arc(Graphe g, unsigned int u, unsigned int v, double val);*

Fonction permettant d’ajouter un arc de sommet de départ u et de sommet d’arrivée v et ayant pour poids val.

* Lors de ce projet, il faut aussi écrire des fonctions correspondant à notre objectif.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Graphe lit\_graphe(char\* fichier);

Fonction qui à partir d’un fichier .csv, crée un graphe et le remplit avec les données du fichier. Cette fonctione est complexe car elle doit utiliser lesfonctions fscanf et fgets et remplir le graphe. Elle utilise donc des focntions que nous avons décrite plus tot et qui doivent donc fonctionner paraitement pour que cette fontion fonctionne.

double graphe\_pcc(Graphe g, unsigned int u, unsigned int v);

Cette fonction est le restranscription en langage C de l’algorithme de plus court chemin de Belleman ci-dessous. Elle fera par la suite l’objet d’une optimisation.

///////////////////insérer algorithme de Belleman///////////////

Les Tests

Durant ce projet, différents test vont être réalisés. Ces tests sont de deux types :

-Les tests unaires permettront de tester une fonction indépendemment des autres. Ces tests porteront surtout sur les fonctions de la structure graphe et la fonction de calcul de plus court chemin. Quatre tests seront ainsi réalisés.

-Les tests «  »

Groupes de fichier(makefile, .c,.h,etc)

Il y a donc dans ce projet 4 groupes de fichier :

-Les fichiers déjà réalisés en séance 9 et 10 portant sur les listes :

-liste.h

-liste.c

-element.h

-les fichiers des graphes :

-graphe.c

-graphe.h

- les fichiers tests : tests1.c à tests6.c

-le Make file et le main

Fonctions de graphe.c :

Tests :

Fichier main() :

Make file :

Répartition et planning du projet