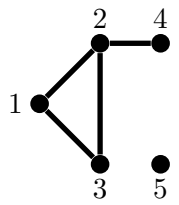
INF233-TP3 : Manipulation de graphes

8 novembre 2011

Le but de ce TP est de savoir faire des manipulations basiques sur des graphes. Deux représentations seront utilisées :
- par matrice d'adjacence : matrice carrée binaire M de taille $n \times n$ (n = ordre du graphe) où $M_{ij} = 1$ si et seulement si ij est une arête du graphe. La matrice est alors symétrique avec la diagonale remplie de 0.
- par matrice d'incidence : matrice binaire M de taille $n \times m$ (m = nombre d'arêtes) où $M_{ie} = 1$ si et seulement si i est une extrémité de e.

On représentera les matrices par des tableaux doubles. Pour plus de simplicité, on définiera deux constantes $NMAX$ et $MMAX$ en début de fichier pour indiquer respectivement le nombre maximum de sommets et d'arêtes dans les graphes considérés. On utilisera ensuite seulement une petite partie des matrices, dont la taille sera donnée dans les arguments des fonctions. Il pourra aussi être plus aisé de ne pas considérer la case 0 des tableaux.

Pour tester les fonctions, nous considérerons l'exemple suivant :



1 Autour des différentes représentations des graphes

Question 1 – Ecrivez à la main la matrice d'adjacence du graphe dessiné puis écrire une fonction `void exemplebase(int G[NMAX][NMAX])` qui initialise le tableau G donné en entrée avec la matrice d'adjacence de l'exemple.

Question 2 – Ecrivez une fonction `void exemplehasard(int G[NMAX][NMAX], int n)` qui initialise le tableau G avec une matrice d'adjacence aléatoire de taille $n \times n$. (Attention, la matrice doit avoir sa diagonale nulle et ne contenir que des 1 et des 0).

Indice : Pour générer des nombres au hasard, on pourra utiliser la fonction `int rand(void)` qui renvoie un nombre entier au hasard. Pensez à inclure la librairie `stdlib.c`. (En relançant votre programme, que remarquez-vous ? Vous pouvez pour réparer cela utiliser la commande `srand(time(NULL));`, à placer au début de la fonction main.)

Question 3 – Ecrivez une fonction `void afficher(int tab[NMAX][NMAX], int n)` qui affiche le tableau pour les indices allant de 1 à n.

Question 4 – Ecrivez une fonction `int nbaretes(int G[NMAX][NMAX], int n)` qui renvoit le nombre d'arêtes du graphe représenté par la matrice d'adjacence G, et d'ordre n, donné en argument. Testez votre fonction avec l'exemple de base et avec une matrice tirée au hasard.

Question 5 – Ecrivez une fonction `void adjversincid(int Madj[NMAX][NMAX], int Minc[NMAX][MMAX], int n)` qui transforme la matrice d'adjacence d'un graphe (contenu dans Madj) en matrice d'incidence (à stocker dans Minc).

Combien d'opérations élémentaires sont effectuées (donnez un ordre de grandeur) ? Testez votre fonction en affichant la matrice d'incidence ainsi obtenue.

Question 6 – Ecrivez une fonction `void incidversadj(int Minc[NMAX][NMAX], int Madj[NMAX][MMAX], int n, int m)` qui transforme la matrice d'incidence d'un graphe (contenu dans Minc) en matrice d'adjacence (à stocker dans Madj).

Combien d'opérations élémentaires sont effectuées (donnez un ordre de grandeur) ?

Testez votre fonction. Verifiez qu'en appliquant à la suite les fonctions `adjversincid` et `incidversadj` on retrouve bien la matrice de départ. Que se passe-t-il dans le cas inverse ?

Question 7 – Ecrire une fonction qui affiche la représentation par liste d'adjacence à partir de la matrice d'adjacence d'un graphe.

2 Autour du voisinage et des degrés

Question 8 – Ecrivez une fonction qui dit si deux sommets sont voisins. Faites-le pour les deux cas (matrice d'incidence et matrice d'adjacence).

On appelle *degré* d'un sommet son nombre de voisins.

Question 9 – Ecrivez une fonction qui calcul le degré d'un sommet donné en argument. Faites-le pour les deux cas (matrice d'incidence et matrice d'adjacence).

Question 10 – Ecrivez une fonction qui dit si un sommet est isolé. Choisissez la représentation que vous voulez.

Question 11 – Ecrivez une fonction qui remplit un tableau `Degres` tel que `Degres[i]` contiennent le degré de i apres appel à la fonction.

Question 12 – Ecrire une fonction qui calcule la somme des degrés de tous les sommets du graphe. A quoi cette somme est-elle égale (par rapport à d'autres paramètres du graphe) ? Vérifiez le sur plusieurs exemples.

Question 13 – Ecrivez une fonction qui renvoit le numéro du sommet avec le plus grand degré.

Question 14 – Ecrivez une fonction qui donne les numéros des sommets classés par degré décroissant. On pourra stocker le résultat dans un tableau.

Question 15 – Ecrivez une fonction qui teste si deux sommets i et j ont un voisin en commun. La fonction renvoit 0 si les deux sommets n'ont pas de voisins en commun et sinon, renvoit le numéro du voisin commun. (Choisir la représentation que vous préférez).

Question 16 – Ecrivez une fonction qui prend un ensemble de sommets en entrée (représenté par exemple par un tableau de booléen de taille n) et qui renvoit 1 s'il n'y a aucune arête entre deux sommets de l'ensemble, 0 sinon. Testez votre fonction.