Algorithmique des Graphes – TD 1

Exercice 1

Est-il possible qu'un groupe de 9 personnes soit tel que chacun soit l'ami de 5 personnes exactement du groupe ?

Exercice 2

Un graphe simple non orienté G a 15 arêtes, 3 sommets de degré 4, les autres étant de degré 3. Quel est le nombre de sommets du graphe ?

Exercice 3

Montrer que dans un graphe non orienté simple il y a au moins deux sommets de même degré.

Exercice 4

Pour le graphe orienté ci-dessous donner les différentes représentations vues en cours.

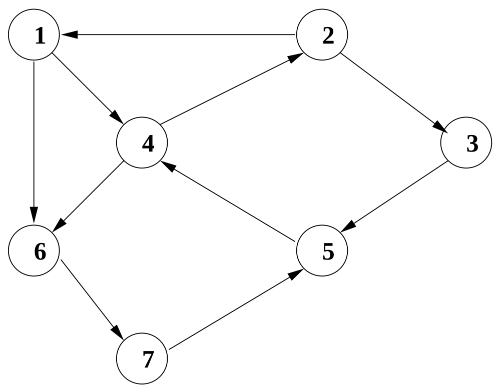

Fig. 1 – Graphe G1

Exercice 5

Dessiner le graphe défini par la matrice d'adjacence :

$$\begin{vmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{vmatrix}$$

Exercice 6

Appliquer l'algorithme du parcours en largeur aux graphes des deux exercices précédents à partir du sommet 1.

```
Parcours_en_Largeur(G,x0)
{
 Créer_File(f);
 pour chaque sommet u de G faire
      visité[u] = false;
 visité[x0] = true;
 Ajouter(x0,f);
 tant que (! Vide(f))
    {
      x = Tête(f);
      Supprimer(f);
      pour toute extrémité y d'un arc d'origine x faire
            si (!visité[y])
               {
                 visité[y] = true;
                 Ajouter(y,f);
               }
    }
}
```

Exercice 7

On se donne un graphe orienté à n sommets et m arcs. On suppose que les degrés d^+ et d^- de chaque sommet valent au plus d.

Donner une borne supérieure pour la complexité de chacune des opérations suivantes :

- faire afficher tous les successeurs d'un sommet donné.
- faire afficher tous les prédécesseurs d'un sommet donné.
- déterminer si un graphe est sans boucle (une boucle est un arc $s \to s$).
- parcours en largeur à partir d'un sommet donné.

suivant qu'on utilise, pour représenter le graphe, une matrice d'adjacence, des listes d'arcs, des listes d'adjacence, ...

Exercice 8

Un graphe non orienté est *biparti* si l'on peut partager l'ensemble de ses sommets en deux sous-ensembles tels qu'aucune arête du graphe relie deux sommets appartenant au même sous-ensemble.

Proposer un algorithme efficace permettant de déterminer si un graphe non orienté est biparti. Quelle est sa complexité ?