# TP1- Connexité, Matrices d'adjacence d'un graphe

Objectifs: Manipuler les matrices d'adjacence

On suppose *N* villes avec une matrice des distances donnée. Deux villes sont reliées par voie radio si elles sont éloignées de moins de *d* kms

Le listing suivant contient les fragments d'un script Matlab. Complétez le en répondant aux questions suivantes et en ajoutant les commandes correspondantes dans le script *TP1.m*.

```
addpath('matlab_bgl');      %load graph libraries
addpath('matlab_tpgraphe'); %load tp ressources
load TPgraphe.mat;          %load data

%%%%%% DISPLAY INPUT DATA ON TERMINAL %%%%%
cities %names of cities
D      %distance matrix bw cities
pos    %x-y pos of the cities
```

## Exercice 1

**Q1)** Modéliser le graphe *A*, représentant la connectivité entre les villes pour une portée radio de 500km.
La fonction *viz_adj (D, A ,pos , cities)* permet d'afficher le graphe *A*.

La fermeture transitive d'un graphe *G*(*X, A*) est la relation transitive minimale contenant la relation (*X, A*), il s'agit d'un graphe *G**=(*X,* *) tel que (*x, y*) ∈* si et seulement s'il existe un chemin *f* dans *G* d'origine *x* et d'extrémité *y*.

La fonction *graphPower (G, n)* calcule de façon itérative la fermeture transitive. Afficher la matrice de retour de *graphPower* pour les valeurs de *n* = 2, 3, 10 et 12. Cette matrice indique la présence de chemins de longueur <=*n*.

**Q2)** Afficher en utilisant les fonctions *viz_adj* et *graphPower* les matrices de présence de chemins de longueur respectivement inférieurs à 2, 3, 10 et 12. Que constatez vous ? Expliquer.

```
%%%%%%EXO 1 (modeliser et afficher le graphe) %%%%%
A= XXX à faire %adj matrix
viz_adj(D,A,pos,cities);
viz_adj(D,XXX à faire,pos,cities);
```

## Exercice 2

On cherche à déterminer pour ces villes :

**Q1)** l'existence d'un chemin d'interconnexions de 3 sauts exactement.
**Q2)** le nombre de chemins d'interconnexions de 3 sauts exactement.
**Q3)** le nombre de chemins d'interconnexions de longueur inférieure ou égale à 3 sauts.

On utilisera *bmul(A,B)* pour la multiplication booléenne de matrices.

```
%%%%%% EXO 2 %%%%%
%Q1 - existence d'un chemin de longueur 3
```

```
XXX à faire
%Q2 - nb de chemins de 3 sauts
XXX à faire
%Q3 - nb de chemins <=3
XXX à faire
```

## Exercice 3

On stocke une chaîne *c* comme une suite de sommets (vecteur d'indices des sommets).

**Q1)** Quelles sont les conditions sur les paires successives des sommets d'une chaîne pour que celle-ci appartienne au graphe ?

**Q2)** Ecrire la fonction *possedechaine(G, chaine)* qui permet de déterminer si une chaîne donnée appartient à un graphe *G*.

**Q3)** Tester les chaînes : « *Paris, Londres, Dublin* », « *Paris, Bernes, Ankara* » et « *Zagreb, Berlin, Oslo* »

```
%%%%%% EXO 3 %%%%%
c = [18 13 9]; %la chaine 18 13 9 est elle dans le graphe?
possedechaine(A,c)
c = [18 6 3]; %la chaine 18 6 3 est elle dans le graphe?
possedechaine(A,c)
C = [26 5 17]; %la chaine 26 5 17 est elle dans le graphe?
possedechaine(A,c)
```

## Exercice 4

Une chaîne eulérienne est une chaîne empruntant une fois et une fois seulement chaque arrête du graphe.

**Q1)** Implémenter à l'aide des fonctions matricielles une méthode (*isEulerien*) pour déterminer si un graphe possède une chaîne eulérienne.

**Q2)** Tester sur le graphe *G* de matrice d'adjacence *A*. Commenter le résultat.

```
%%%%%%% EXO 4 %%%%%
isEulerien(A)
```

## Exercice 5

On cherche à identifier les valeurs de d pour lequel le graphe admet une chaîne eulérienne.

Proposer une méthode qui détermine pour toutes les valeurs de portée *d* (dans *R*) si *G(d)* possède une chaîne eulérienne.

La fonction *porteeEulerien(D)* affiche les valeurs de *d* pour lesquelles *G(d)* admet une chaîne Eulérienne. Invoquer la fonction portéeEulerien sur D et expliquer.

```
%%%%%%% EXO 5 %%%%%
porteeEulerien(D)
```