# Théorie des Graphes
## Connexité et Chemins

TPS 1A INOC

theoleyre@unistra.fr

---

*Objectif* : propriétés fondamentales de connexité et de recherche de chemins

*Durée* : 2 heures

---

Partie 1

## Graphes non orientés

### Exercice 1: Connexité et transitivité

Quelle propriété satisfait la relation d'un graphe non orienté ?
Montrer qu'un graphe $G = (S, A)$ est connexe si et seulement si sa fermeture transitive est le graphe complet d'ordre $|S|$.

### Exercice 2: Cycles et cardinalités

Soit $G = (S, A)$ un graphe non orienté à $n$ sommets.

1. Soit $n \geq 1$. Démontrer que si $G$ est connexe, alors $Card(A) \geq n - 1$.
2. Soit $G$ complet. Combien possède-t-il d'arêtes ? Démontrer ce résultat.
3. Démontrer que si $G$ est sans cycle, alors $Card(A) \leq n - 1$.

### Exercice 3: Graphe biparti

Soit n,m des entiers naturels non nuls. Le graphe complet biparti $K_{n,m}$ est le graphe dont l'ensemble des sommets est partitionné en deux ensembles $V_1$ et $V_2$ de cardinalité respective n et m, et dont les arêtes sont $(u, v)$, avec $u \in V_1$, $v \in V_2$.

1. Quel est le nombre d'arêtes de $K_{n,m}$ ?
2. $K_{n,m}$ est-il connexe ? Quel est son diamètre ? Quels sont les centres ?
3. Sous quelles conditions $K_{n,m}$ est-il Eulérien ?
4. Donnez une condition nécessaire (non triviale) pour que $K_{n,m}$ soit Hamiltonien

## Exercice 4: La chèvre, le loup et le chou

Un passeur souhaite transporter sur la rivière une chèvre, un loup, un chou. La barque ne comprend que 2 places (lui compris) et il ne peut pas laisser sans surveillance sur la même berge le loup et la chèvre, ainsi que le chou et la chèvre.

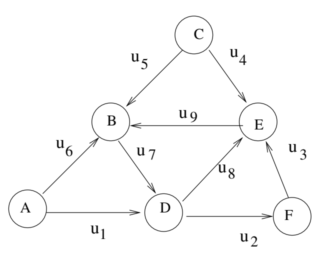
Pouvez vous l'aider ?

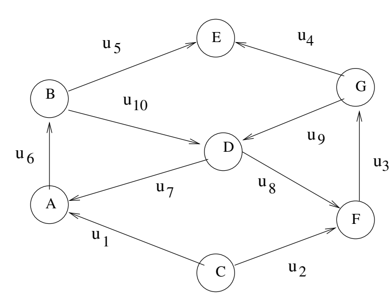
Partie 2

# Graphes orientés

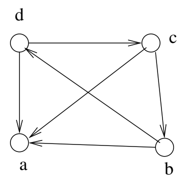
## Exercice 1: Chemins et connexité

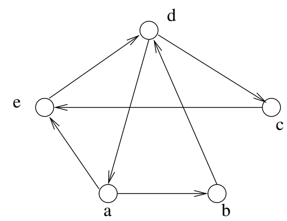
1. On considère le graphe orienté ci-après (gauche).
   (a) Donner un chemin simple de A à F passant par E.
   (b) Donner un chemin élémentaire de A à F.
2. On considère le graphe orienté ci-après (droite).
   (a) Donner un circuit partant de A et passant par G.
   (b) Donner un circuit élémentaire à partir de A.

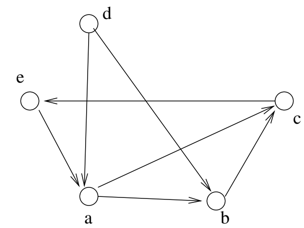


3. Donner les composantes fortement connexes des graphes orientés représentés ci-après.








## Exercice 2: Démonstration

Démontrer la proposition suivante. Soit $G = (S, A)$ un graphe orienté. La relation $\sim_G$ définie sur $S$ par $x \sim_G y \Leftrightarrow$ « il existe un chemin de $x$ à $y$ dans $G$ et un chemin de $y$ à $x$ dans $G$ » est une relation d'équivalence sur $S$.

Partie 3

# Connexité, Chemin, Coupe

## Exercice 1: Labyrinthe

Un labyrinthe est composé de salles reliées entre elles par des portes. L'une des salles est l'entrée du labyrinthe, une autre est la sortie.

1. Modéliser un tel labyrinthe sous forme de graphe.
2. Donner un algorithme permettant :
   (a) de savoir s'il est possible de sortir du labyrinthe ;
   (b) si oui, de calculer un chemin de la salle d'entrée à la salle de sortie.
3. Montrer que l'algorithme proposé termine et est correct.

## Exercice 2: Coûts au pire

1. Considérer l'algorithme proposé dans l'exercice 4 du premier TD, calculant la fermeture transitive d'une relation d'un graphe.

(a) Dans le cas d'un graphe non orienté $G = (S, A)$, avec $|S| = n$ et $|A| = m$, exprimer le coût au pire de cet algorithme en fonction de $n$ et $m$.

(b) Adapter cet algorithme pour résoudre la question 2. (a) de l'exercice 6.

2. Calculer le coût au pire de l'algorithme proposé dans l'exercice 6.
3. Quelle différence entre ces deux procédures justifie une telle différence en coût au pire ?

Tester si une relation est **réflexive** :
*fonction :* estréflexive
*paramètres :* $R$ matrice d'adjacence de $\mathcal{R}$
*résultat :* Vrai ou Faux.
entier $i = 1$ ;
retour$\leftarrow \neg(R(i,i) = 0)$ ;
tant que retour et $i \leq n-1$
$i \leftarrow i+1$ ;
retour$\leftarrow \neg(R(i,i) = 0)$ ;
fin tant que
sortir avec la valeur retour ;

Tester si une relation est **antisymétrique** :
*fonction :* estantisymétrique
*paramètres :* $R$ matrice d'adjacence de $\mathcal{R}$
*résultat :* Vrai ou Faux.
entiers $i = 1$, $j = 2$ ;
retour$\leftarrow \neg((R(i,j) \neq 0) \wedge (R(j,i) \neq 0))$ ;
tant que retour et $i \leq n-1$
tant que retour et $j \leq n-1$
$j \leftarrow j+1$ ;
retour$\leftarrow \neg((R(i,j) \neq 0) \wedge (R(j,i) \neq 0))$ ;
fin tant que
$i \leftarrow i+1$ ;
$j \leftarrow i+1$ ;
fin tant que
sortir avec la valeur de retour ;

Tester si une relation est **symétrique** :
*fonction :* estsymétrique
*paramètres :* $R$ matrice d'adjacence de $\mathcal{R}$
*résultat :* Vrai ou Faux.
matrice $R'$ =transposée($R$) ;
sortir avec la valeur de $R - R' = 0_{\mathcal{M}_{n,n}(\mathbb{N})}$;

Tester si une relation est une **relation d'ordre** :
*fonction :* estordre
*paramètres :* $R$ matrice d'adjacence de $\mathcal{R}$
*résultat :* Vrai ou Faux.
sortir avec la valeur estréflexive($R$)$\wedge$estantisymétrique($R$)$\wedge$esttransitive($R$) ;

Calculer la **fermeture transitive** d'une relation :
*fonction :* fermeture
*paramètres :* $R$ matrice d'adjacence de $\mathcal{R}$
*résultat :* $\mathring{R}$ matrice d'adjacence de $\mathring{\mathcal{R}}$
matrices $R_{temp} \leftarrow R$, $\mathring{R} \leftarrow R * R + R$ ;
tant que $\neg(\mathring{R} \subseteq R_{temp})$
$R_{temp} \leftarrow \mathring{R}$ ;
$\mathring{R} \leftarrow R_{temp} * R_{temp} + R_{temp}$ ;
fin tant que
sortir avec la valeur de $\mathring{R}$ ;

Tester si une relation est **transitive** :
*fonction :* esttransitive
*paramètres :* $R$ matrice d'adjacence de $\mathcal{R}$
*résultat :* Vrai ou Faux.
sortir avec la valeur ($R \subseteq$fermeture($R$)) ;

**Remarque :** Dans la fonction fermeture, l'instruction ($\mathring{R} \subseteq R_{temp}$) signifie $\mathring{R}(i,j) \neq 0 \Rightarrow R_{temp}(i,j) \neq 0$. Mais grâce à l'instruction $\mathring{R} \leftarrow R_{temp} * R_{temp} + R_{temp}$, c'est ici équivalent à : $\forall (i,j) \in E^2, (R_{temp}(i,j) = 0) \Leftrightarrow (\mathring{R}(i,j) = 0)$. Il en va de même dans la fonction esttransitive.