Chapitre 1 : Notions fondamentales de la théorie des graphes

· Définitions d'un graphes et différentes représentations

. Graphes particuliers : Graphes planaires, Graphe dual, Graphes aux arêtes...

· Applications

Chapitre 2 : Arbres et Arborescence

· Introduction

· Arbre couvrant arbre

· Algorithme de Prim

· Algorithme de Kruskall

Chapitre 3 : Problèmes du plus court chemin

· Introduction au problème du plus court chemin

· Algorithme de Dijkstra

· Algorithme de Bellman-Ford

· Algorithme de Floyd-Warshall

Chapitre 4 : Problèmes de flots

· Définitions

· Cycles élémentaires et flots élémentaires

· Problème du flot maximal dans un réseau de transport

· Algorithme de recherche du flot maximal (Ford-Fulkerson)

Chapitre 5 : Méthodes d'ordonnancement

· Diagramme de Gantt

· Méthode PERT

Références*(Livres et polycopiés, sites internet, etc.)* :

- Christian Prins : Algorithmes de graphes (avec programmes en Pascal) Eyrolles, Paris, 1994.
- Bernard Roy : Algèbre moderne et théorie des graphes TomeII, Dunod, 1989
- Graphes et Algorithmes, Eyrolles, Paris 1984

Chapitre 1 : Notions fondamentales de la théorie des graphes

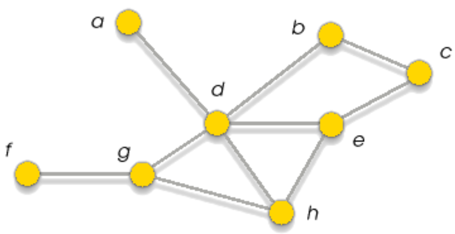
Un graphe permet de décrire un ensemble d'objets et leurs relations, c'est à dire les liens (arêtes) entre les objets (nœuds/sommets).

I. Graphe non orienté

I.1. Définition: Un graphe non orienté G est un couple (V,E).

- V est un ensemble (fini) d'objets. Les éléments de V sont appelés les sommets du graphe.
- E est sous-ensemble de VxV. Les éléments de E sont appelés les arêtes du graphe. Une arête e du graphe est une paire e=(x,y) de sommets. Les sommets x et y sont les extrémités de l'arête.

Exemple



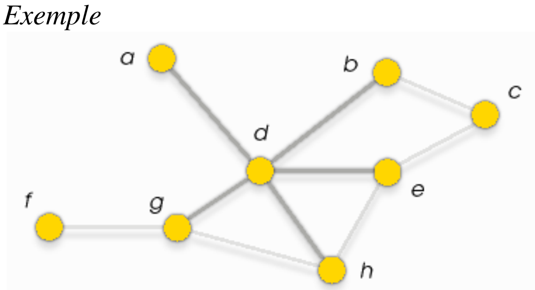
E={ }

V={ }

I.2. Degré d'un sommet/graphe

- Deux sommets ***x*** et ***y*** sont **adjacents** s'il existe l'arête ***(x,y)*** dans ***E***. Les sommets ***x*** et ***y*** sont alors dits **voisins**
- Une arête est **incidente** à un sommet ***x*** si ***x*** est l'une de ses extrémités.
- Le **degré** d'un sommet ***x*** de ***G*** est le nombre d'arêtes incidentes à ***x***. Il est noté **d*(x)***.
- Pour un graphe simple le degré de ***x*** correspond également au nombre de sommets adjacents à ***x***.

Exemple



Pour un graphe simple d'ordre ***n,*** le degré d'un sommet est un entier compris entre ***0*** et ***n-1***. Un sommet de degré ***0*** est dit ***isolé*** : il n'est relié à aucun autre sommet.

Propriété1: La somme des degrés des sommets d'un graphe est égal à 2 fois son nombre d'arêtes.

Propriété2: Le nombre de sommets de degré impair d'un graphe est pair.

- Le **degré d'un graphe** est le degré maximum de tous ses sommets.
- Un graphe dont tous les sommets ont le même degré est dit **régulier**. Si le degré commun est *k*, alors on dit que le graphe est *k*-régulier.

I.3. Sous-graphe et graphe partiel

Pour caractériser de manière moins locale la structure d'un graphe, il est possible de rechercher des parties remarquables du graphe, en restreignant soit l'ensemble des sommets (sous-graphe), soit l'ensemble des arêtes (graphe partiel).

- Un *sous-graphe* de ***G*** consiste à considérer seulement une partie des sommets de ***V*** et les liens induits par ***E***. Par exemple si ***G*** représente les liaisons aériennes journalières entre les principales villes du monde, un sous-graphe possible est de se restreinte aux liaisons journalières entre les principales villes européennes.

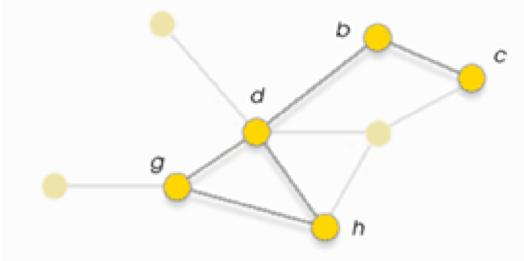
- Un *graphe partiel* de ***G*** consiste à ne considérer qu'une partie des arêtes de ***E***. En reprenant le même exemple, un graphe partiel possible est de ne considérer que les liaisons journalières de moins de 3 heures entre les principales villes du monde.

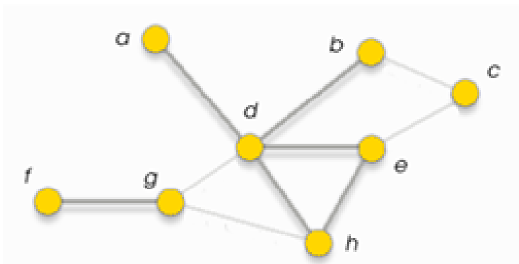
Définition: Pour un graphe ***G = (V,E)***

- Un **sous-graphe** de ***G*** est un graphe ***H=(W, E(W))*** tel que ***W*** est un sous-ensemble de ***V***, et ***E(W)*** sont les arêtes induites par ***E*** sur ***W***, c'est à dire les arêtes de ***E*** dont les 2 extrémité sont des sommets de ***W***. $E(W) = \{(x,y) \in E \,/ x,y \in W\}$
- Un **graphe partiel** de ***G*** est un graphe ***I=(V,F)*** tel que ***F*** est un sous ensemble de ***E***.

  Un sous-graphe ***H*** de ***G*** est entièrement défini (induit) par ses sommets ***W***, et un graphe partiel ***I*** par ses arêtes ***F***.

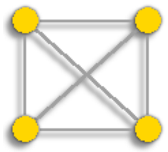
Exemple





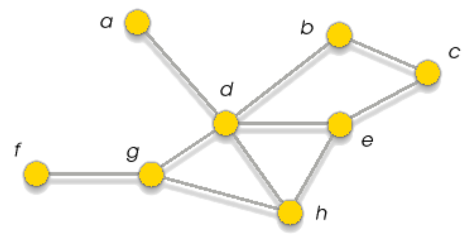
I.4. Graphe complet, clique, stable

Définition: Un **graphe complet** est un graphe où chaque sommet est relié à tous les autres. Le graphe complet d'ordre ***n*** est noté ***Kn***. Dans ce graphe chaque sommet est de degré ***n-1***.



- Une **clique** est un sous-graphe complet.
- Un **stable** est un sous-graphe sans arête.

Exemple



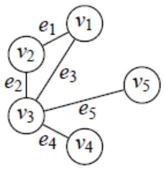
2 cliques d'ordre 3 définies par les ensemble de sommets ***{ d, g, h }*** et ***{d, e, h }***

4 stables d'ordre 4 définis par les ensemble de sommets ***{ a, b, e, g }*** *et* ***{a, b, h, f }***
{ a, c, h, f } *et* ***{ a, b, e, f }***

I.5. Chaine et cycle

Définition: Une **chaîne** dans *G*, est une suite ayant pour éléments alternativement des sommets et des arêtes, commençant et se terminant par un sommet, et telle que chaque arête est encadrée par ses extrémités.
On dira que la chaîne **relie** le premier sommet de la suite au dernier sommet. En plus, on dira que la chaîne a pour longueur le nombre d'arêtes de la chaîne.

Le graphe ci-dessous contient entre autres les chaînes (*v*1, *e*1, *v*2, *e*2, *v*3, *e*5, *v*5)
et (*v*4, *e*4, *v*3, *e*2, *v*2, *e*1, *v*1).



On ne change pas une chaîne en inversant l'ordre des éléments dans la suite correspondante. Ainsi, les chaînes (*v*1, *e*3, *v*3, *e*4, *v*4) et (*v*4, *e*4, *v*3, *e*3, *v*1) sont identiques.

Définitions

- On appelle **distance** entre deux sommets la longueur de la plus petite chaîne les reliant.
- On appelle **diamètre** d'un graphe la plus longue des distances entre deux sommets.
- Une chaîne est **élémentaire** si chaque sommet y apparaît au plus une fois.
- Une chaîne est **simple** si chaque arête apparaît au plus une fois.

  Dans le graphe précédent, (v1, e1, v2, e2, v3) est une chaîne simple et élémentaire.
- Une chaîne dont les sommets de départ et de fin sont les mêmes est appelée chaîne **fermée**.

  Dans le graphe précédent, (v4, e4, v3, e5, v5, e5, v3, e4, v4) est une chaîne fermée.
- Une chaîne fermée simple est appelée **cycle**.

  Dans le graphe précédent, la chaîne (v1, e1, v2, e2, v3, e3, v1) est un cycle.

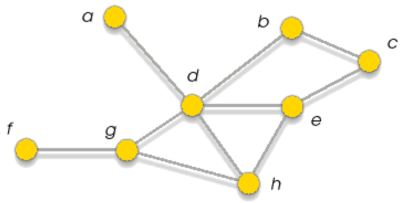
Propriété: Dans un graphe ***G*** d'ordre ***n***

- Toute chaine élémentaire est de longueur au plus ***n*-1**
- Le nombre de chaines élémentaires dans le graphe est fini.

I.6. Connexité d'un graphe

Définition: Un graphe est **connexe** ssi il existe une chaine entre chaque paire de sommets.

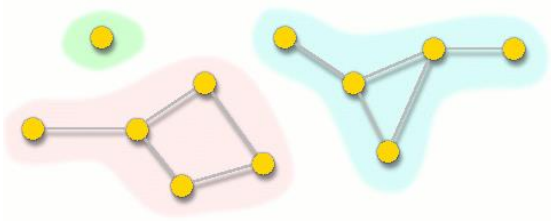
Exemple



Si le graphe ***G*** n'est pas connexe, Il apparaît alors comme un ensemble de graphes connexes mis les uns à coté des autres. Chacun de ces graphes est un sous-graphe particulier de ***G***, appelé ***composante connexe***.

Définition: Une composante connexe d'un graphe ***G*** est un sous-graphe ***G'=(V',E')*** connexe maximal: il n'est pas possible d'ajouter à ***V'*** d'autres sommets en conservant la connexité du sous-graphe.

Exemple



- Un graphe ne possédant qu'une seule composante connexe est simplement un graphe connexe.
- Un sommet isolé (de degré 0) constitue toujours une composante connexe à lui seul.
- La relation sur les sommets "il existe une chaine entre ..." est une relation d'équivalence (réflexive, symétrique et transitive). Les composantes connexes d'un graphe correspondent aux classes d'équivalences de cette relation.

Propriété: Un graphe ***G*** d'ordre ***n*** connexe comporte au moins ***n*-1** arêtes.

I.7. Graphe acyclique: Un graphe dans lequel il n'existe aucun cycle est dit *acyclique*.

Propriété1: Si dans un graphe ***G*** tout sommet est de degré supérieur ou égal à 2, alors ***G*** possède au moins un cycle. Cette propriété simple implique qu'un graphe sans cycle possède au moins un sommet de degré 0 ou 1.

Propriété2: Un graphe acyclique ***G*** à ***n*** sommets possède au plus ***n*-1** arêtes.

Définition: Pour un graphe G ayant m arêtes, n sommets et p composantes connexes, on définit : n $(G) = m-n+ p$. n (G) est appelé le nombre **cyclomatique**.

- On a n $(G) > 0$ pour tout graphe G.
- n $(G) = 0$ si et seulement si G est sans cycle.

II. Graphe orienté

Un digraphe fini $G = (V,E)$ est défini par l'ensemble fini $V = \{v_1, v_2, . . . , v_n\}$ dont les éléments sont appelés sommets, et par l'ensemble fini $E = \{e_1, e_2, . . . , e_m\}$ dont les éléments sont appelés **arcs**.
Un arc e de l'ensemble E est défini par une paire ordonnée de sommets. Lorsque $e=(u, v)$, on dit que l'arc e va de u à v. On dit aussi que u est l'extrémité initiale et v l'extrémité finale de e.

II.1. Degré d'un sommet d'un digraphe

Soit v un sommet d'un graphe orienté. On note $d^+(v)$ le degré **extérieur** du sommet v, c'est-à-dire le nombre d'arcs ayant v comme extrémité initiale.
On note $d^-(v)$ le degré **intérieur** du sommet v, c'est-à-dire le nombre d'arcs ayant v comme extrémité finale. On définit le degré : $d(v) = d^+(v)+d^-(v)$

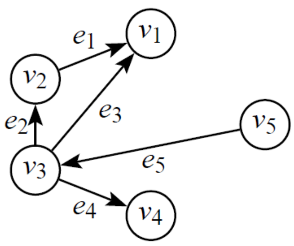
II.2. Chemins et circuits

Un **chemin** conduisant du sommet a au sommet b est une suite ayant pour éléments alternativement des sommets et des arcs, commençant et se terminant par un sommet, et telle que chaque arc est encadré à gauche par son sommet origine et à droite par son sommet destination. On ne peut donc pas prendre les arc à rebours. Par convention, tout chemin comporte au moins un arc.

On appelle **distance** entre deux sommets d'un digraphe la longueur du plus petit chemin les reliant. S'il n'existe pas de chemin entre les sommets x et y, on pose $d(x, y) = \infty$.

Un **circuit** est un chemin dont les sommets de départ et de fin sont les mêmes.

Les notions de chemins et de circuits sont analogues à celles des chaînes et des cycles pour les graphes non orientés.



- exemple le chemin ($v3$, $e2$, $v2$, $e1$, $v1$).
- $d(v5, v4) = 2$, $d(v4, v5) = \infty$, $d(v3, v1) = 1$.
- Le digraphe ci-dessus ne contient pas de circuit.

II.3. Digraphe fortement connexe

Un digraphe est **fortement connexe**, si toute paire ordonnée (a,b) de sommets distincts du graphe est reliée par au moins un chemin. En d'autres termes, tout sommet est atteignable depuis tous les autres sommets par au moins un chemin.
On appelle **composante fortement connexe** tout sous-graphe induit maximal fortement connexe (maximal signifie qu'il n'y a pas de sous-graphe induit connexe plus grand contenant les sommets de la composante).

II.4. Digraphes sans circuit

Théorème
Le digraphe G est sans circuit si et seulement si on peut attribuer un nombre $r(v)$, appelé le **rang** de v, à chaque sommet v de manière que pour tout arc (u, v) de G on ait $r(u) < r(v)$.
Preuve
Si G comporte un circuit C, il n'est pas possible de trouver de tels nombres $r(i)$ car, autrement, considérant $r(j) = \max\{r(i) \mid i \in C\}$ et l'arc $(j, k) \in C$, on aurait $r(j) \leq r(k)$ en contradiction avec la définition du rang.
Réciproquement, si G n'a pas de circuit, il existe au moins un sommet sans prédécesseur dans G (sans cela, en remontant successivement d'un sommet à un prédécesseur, on finirait par fermer un circuit). Ainsi, on peut attribuer séquentiellement des valeurs aux sommets du graphe à l'aide de l'algorithme qui suit, ce qui conclura la démonstration.

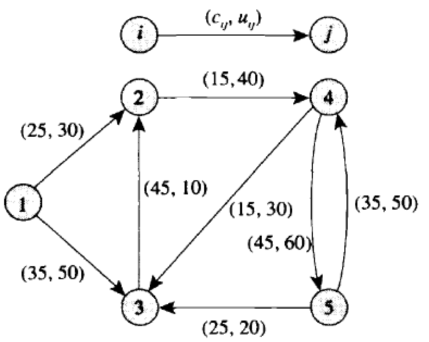
Algorithme de calcul du rang
Donnée : digraphe $G = (V,E)$ sans circuit.
Résultat : rang $r(v)$ de chaque sommet $v \in V$ du digraphe G.
Début
$r := 0$
$X := V$
R : l'ensemble des sommets de X sans prédécesseur dans X
Tant que X n'est pas vide **faire**
$r(v) := r$ pour tout sommet $v \in R$
$X := X - R$
R : l'ensemble des sommets de X sans prédécesseur dans X
$r := r+1$
Fin tant que
Fin

III. Représentation de graphes

La performance d`un algorithme de graphe, ne dépend pas seulement de l`algorithme , mais aussi de la manière utilisée pour représenter le graphe en mémoire. En représentant un graphe, nous avons besoin de stocker deux types d`informations: *(1)* la topologie du graphe, i.e, la structure des nœuds et des arcs du graphe, *(2)* des données telles que les couts, les capacités ...

**III.1. Matrice d`incidence (nœud-arc):** Cette représentation stocke le graphe sous forme de matrice de taille $n$ x $m$: une ligne pour chaque nœud, et une colonne pour chaque arc. La colonne correspondante ã l`arc (i,j) a $+1$ dans la ligne correspondant au nœud i et -1 au rang correspondant au nœud j

Exemple



$$\begin{array}{cc} & \begin{array}{cccccccc} (1,2) & (1,3) & (2,4) & (3,2) & (4,3) & (4,5) & (5,3) & (5,4) \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} & \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & 1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 & 1 \end{bmatrix} \end{array}$$

Dans la matrice d`incidence:

- Seulement 2xm des nxm entrées sont des valeurs non nulles
- Toutes les valeurs non nulles sont égales ã +1 ou -1
- Chaque colonne a exactement un seul +1 et un seul -1.
- Le nombre de +1 dans une ligne est égal du degré sortant du nœud correspondant et le nombre de -1 est égal au degré entrant du nœud.

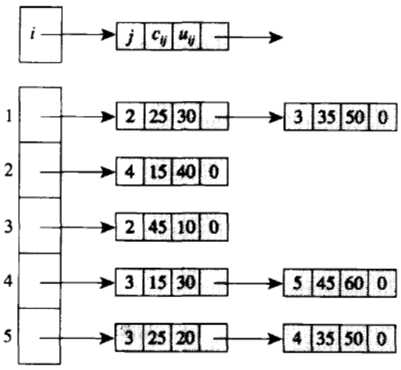
Cette représentation n`est pas efficace en espace mémoire car elle contient peu de coefficients non nuls.

**III.2. Matrice d`adjacence (nœud-nœud):** C`est une matrice nxn. son ij^{eme} élément est égal ã 1 si l`arc (i,j) $\in$A et égal ã 0 sinon. Si nous voulons stocker les couts des arcs et leurs capacites aussi bien que la topologie du graphe, nous pouvons utiliser deux matrices de taille nxn.

La matrice d`adjacence a n$^2$ éléments, seulement m d`entre elles sont non nulles. Cette représentation est donc efficace en espace mémoire seulement si le réseau est suffisamment dense. Cependant la simplicité de cette représentation permet une implémentation facile des algorithmes.

$$
\begin{array}{c|ccccc|}
 & 1 & 2 & 3 & 4 & 5 \\
\hline
1 & 0 & 1 & 1 & 0 & 0 \\
2 & 0 & 0 & 0 & 1 & 0 \\
3 & 0 & 1 & 0 & 0 & 0 \\
4 & 0 & 0 & 1 & 0 & 1 \\
5 & 0 & 0 & 1 & 1 & 0 \\
\hline
\end{array}
$$

III.3. Listes d`adjacence:
La liste d`adjacence d`arcs A(i) d`un nœud i est l`ensemble des arcs émanant de ce nœud i.e la liste des arcs (i,j) $\in$A. Similairement, la liste d`adjacence de nœuds d`un nœud i est l`ensemble des nœuds j tels que (i,j) $\in$A.
La représentation de la liste d`adjacence stocke la liste d`adjacence de nœud de chaque nœud comme une liste chainée. Comme nous devons stocker n listes chainées (une pour chaque nœud), nous avons besoin d`un tableau de pointeurs pour pointer au premier éléments de chaque liste.



III.4. les arcs parallèles: Dans le cas ou deux nœuds sont liés par plus d`un arc, la notation (i,j) ne spécifie plus un arc unique. Pour les graphes avec des arcs parallèles, la représentation de liste d`adjacence est capable de les manipuler. Si un nœud *i* a deux arcs sortants, la liste chainée contiendra deux éléments correspondants a ces deux arcs.

III.5. les graphes non orientés: Nous pouvons représenter les graphes non orientés avec les même représentations précédentes. Cependant:

- Quand un arc (i,j) appartient a un graphe non orienté, les deux paires (i,j) et (j,i) doivent être incluses dans la représentation. Donc l`arc (i,j) est stocké deux fois dans les listes d`adjacence, une fois dans celle de i et une fois dans celle de j.
- D`autres modifications évidentes sont nécessaires. par exemple, dans la matrice d`incidence, la colonne correspondante a l`arc (i,j) aura +1 dans les deux lignes i et j
- La matrice d`adjacence aura +1 dans la position h_{ij} et h_{ji} pour chaque arc (i,j) $\in$A. Et comme la matrice sera symétrique, on peut stocker sa moitié.
- Quand une mise a jour est faite pour un arc, nous devons aussi mettre a jour l`autre arc aussi.

IV. Graphes particuliers

IV.1. Graphe eulérien

- On appelle **cycle eulérien** d'un graphe G un cycle passant une et une seule fois par chacune des arêtes de G.
- Un graphe est dit **eulérien** s'il possède un cycle eulérien.

Plus simplement, on peut dire qu'un graphe est eulérien s'il est possible de dessiner le graphe sans lever le crayon et sans passer deux fois sur la même arête.

IV.2. Graphe hamiltonien

- On appelle **cycle hamiltonien** d'un graphe G un cycle passant une et une seule fois par chacun des sommets de G.
- Un graphe est dit **hamiltonien** s'il possède un cycle hamiltonien.

Propriétés

- un graphe possédant un sommet de degré 1 ne peut pas être hamiltonien
- si un sommet dans un graphe est de degré 2, alors les deux arêtes incidentes à ce sommet doivent faire partie du cycle hamiltonien
- les graphes complets sont hamiltoniens.

Théorème

Soit G un graphe simple d'ordre $n \geq 3$. Si pour toute paire $\{x, y\}$ de sommets non adjacents, on a $d(x)+d(y) \geq n$, alors G est hamiltonien.

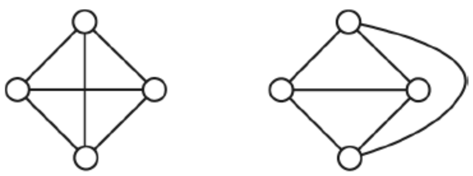
Corollaire

Soit G un graphe simple d'ordre $n \geq 3$. Si pour tout sommet x de G, on a $d(x) \geq n/2$, alors G est hamiltonien.

En effet, un tel graphe vérifie les conditions du théorème précédent, car si x et y ne sont pas adjacents, on a bien : $d(x)+d(y) \geq n/2+n/2=n$

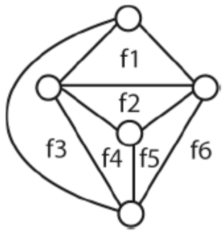
IV.3. Graphes planaires

On dit qu'un graphe est **planaire** si on peut le dessiner dans le plan de sorte que ses arêtes ne se croisent pas.



Définition

Dans un graphe planaire topologique, les zones délimitées par des arêtes qui les entourent sont appelées des « faces ».



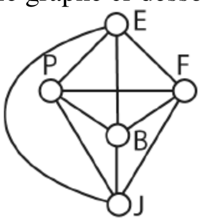
Soit S le nombre de sommets d'un graphe planaire topologique, A son nombre d'arêtes et F son nombre de faces.

Propriété 1
Dans tout graphe planaire topologique connexe, on a F = A – S + 2.

Propriété 2
Tout graphe planaire topologique connexe avec S sommets et dont la plus petite face comporte P arêtes contient au plus (S-2) p/(p-2) arêtes.

Propriété 3
Tout graphe planaire connexe avec S sommets contient au plus 3(S-2) arêtes.

Exemple
Le graphe ci-dessous, ne peut pas être planaire puisque 3(S-2) vaut 9 alors que A vaut 10.



Supposons maintenant qu'aucune face d'un graphe planaire n'est un triangle En d'autres termes, supposons qu'en choisissant trois sommets quelconques dans le graphe, il manque toujours au moins une arête entre deux de ces trois sommets. On a alors P >= 4 et la borne sur le nombre d'arêtes devient donc encore plus petite.

***Propriété* 4 (Manori)**
Tout graphe planaire connexe avec S sommets et sans triangle contient au plus 2(S-2) arêtes.

IV.4. Graphes biparties

Un *graphe biparti* est un graphe dont l'ensemble des sommets peut être partitionné en deux parties *X* et *Y* telles que toute arête a une extrémité dans *X* et l'autre dans *Y*.

Exemple d`application

un ensemble X ={x_1,x_2,...,x_n} de candidats
un ensemble Y={y_1,y_2,...,y_m} d`emplois
$S_i \subseteq$ Y les emplois pour lesquels xi est compétent

On cherche a trouver une affectation d`emplois qui minimise le nombre de sans-emplois
G=(V,E) : $V=X\cup Y$, $(x_i,y_j)\in E$ ssi $y_j\in S_i$

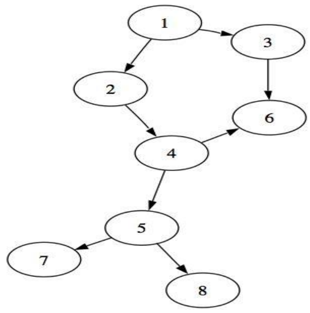
Définition

Une *coloration* ***propre*** de *G* est une fonction associant à tout sommet de *G* une couleur, telle que deux sommets adjacents n'ont pas la même couleur.
Un graphe *G* est *k*-coloriable s'il existe une coloration propre de *G* avec *k* couleurs.

- Un graphe est biparti s'il est 2-coloriable.
- Un graphe est biparti si et seulement si il ne contient pas de cycles de longueur impaire.

V. Parcours des graphes: il existe deux parcours d'un graphe, un type de parcours consiste à explorer le graphe en passant par tous les sommets. Ce type de parcours est un parcours dit hamiltonien. L'autre type de parcours est un parcours dit Eulérien et consiste à emprunter toutes les arêtes du graphe.
Les parcours peuvent se faire de deux façons, on a le parcours en largeur et le parcours en profondeur.

V.1 Parcours en largeur
Un parcours en largeur de G à partir d'un sommet donné s est une permutation de ses sommets $(p_1 , p_2 , p_3 , p_4 , ... , p_n)$ tel que $p_1 = s$ et $d (s , p_i) <= d (s, p_j)$ pour tout $1<i<j<=n$

<table>
<tr><td></td><td>Le parcours en largeur de ce graphe a partir du sommet 1 donnerait la liste suivante : 1 2 3 4 6 5 7 8. Une autre solution serait la suivante : 1 3 2 6 4 5 8 7.</td></tr>
<tr><td>ParcoursLargeur(Sommet s){
f = CreerFile()
f.enfiler(s)
marquer(s)
TANT-QUE (NON f.vide()) FAIRE
s = f.defiler()
POUR TOUT arc(s,v) FAIRE
SI NonMarqué(v) ALORS
f.enfiler(v)
marquer(v)
FIN SI
FIN POUR
FIN TANTQUE</td><td>ParcoursProfondeur (Sommet s){
Marquer(s)
POUR CHAQUE arc(s,v) FAIRE
SI NonMarqué(v) ALORS
ParcoursProfondeur (v)
FIN-SI
FIN-POUR
}</td></tr>
</table>

V.2 Parcours en profondeur: On obtient un parcours en profondeur de G si, à partir d'un sommet donné s, on explore récursivement en profondeur ses successeurs non encore visités.
Dans l`exemple précédent, le parcours en profondeur a partir du sommet 1 donnerait la liste suivante : 1 2 4 5 7 8 6 3.

VI. Applications: Les applications pratiques sur les graphes sont très diverses :

- Optimisation des réseaux de transport: transports routiers ou transports d'information
- Conception de réseaux électriques, de réseaux de communication
- Analyses des réseaux sociaux (WEB)
- Informatique théorique, compilation, bases de données...,
- Géographie (cartographie), architecture, aménagement du territoire…
- administration: établissement des emplois du temps...
- Planification de travaux: pour déterminer la durée minimale des travaux

  ...