Plan du cours de théorie des graphes.
I.U.T. de Blagnac.

Jean-François Culus
culus@univ-tlse2.fr

2004-2005

Sommaire

Introduction.

Le présent document est un résumé du cours donné à l'IUT de Blagnac durant l'année 2004-2005. En aucun cas il ne prétend être un cours autonome, ou permet de se dispenser du dit cours d'amphi dont il en est le support. A titre d'exemple, les démonstrations des théorèmes seront présentées en amphi mais ne sont pas reprisent ici. De même, certains paragraphes de ce document ne seront pas abordés par faute de temps en amphi. Enfin ce document comporte vraissemblablement de nombreuses coquilles, fautes d'orthographe, et mêmes (en faible nombre j'espère!) des fautes de mathématiques. Vous pouvez me les signaler via mail: *culus@univ-tlse2.fr* et je serai fort heureux de les corriger. De même, vous pouvez me contacter pour toutes questions relatives à ce cours, éclaircissements ou remarques.

Chapitre 1: Eléments de base en théorie des graphes.

a- Premières notions et notations

Définitions: Graphe, sommet, arête, sous-graphe. Un graphe G est un objet mathématique composé d'un ensemble non vide fini de *sommets* $V(G)$ et *d'arêtes* $E(G)$ reliant deux sommets. Une arête e reliant les sommets u et v s'écrit comme étant $e = \{u, v\}$ un ensemble de paires (non ordonné). Notons que l'ensemble $E(G)$ des arêtes représente donc une relation sur l'ensemble $V(G)$, c'est-à-dire que $E(G) \in V(G) \times V(G)$. Ainsi, $G = (V(G), E(G)) \in V(G) \times (V(G) \times V(G))$.

Si $U \subset V(G)$ est un sous-ensemble des sommets de G, il nous est possible de définir le sous-graphe $[U]$ induit par G sur U; l'ensemble des sommets sera alors U, et l'ensemble des arêtes de $[U]$ sera $\{\{x, y\}/x \in U, y \in U\}$

Définitions: voisins, extrémités, incidence, adjacence, voisinage, boucle

Si $e = \{u, v\}$ est une arête de G, alors les sommets u et v seront dits *voisins* dans G. Les sommets u et v seront dits être les extrémités de l'arête e. L'arête $e = \{u, v\}$ sera dite être incidente aux sommets u et v. On définit aussi l'ensemble des arêtes incidentes au sommet u comme étant ...

On note $\Gamma_G(u)$ le *voisinage* du sommet u dans G; donc l'ensemble des sommets de G qui sont voisins du sommet u dans G. Nous omettons l'indice G lorsque cela n'induira aucune difficulté de compréhension; cet indice servant en particulier à distinguer le voisinage du sommet x dans G et dans un sous-graphe $[U]$ de G.

Une boucle sera une arête reliant un sommet à lui-même: $e = \{u, u\}$; généralement, on ne considèrera que des graphes sans boucle.

Paramêtres du graphe: ordre, degré, nombre d'arêtes.

L'ordre d'un graphe G est le nombre n de ses sommets, ie: $n = |V(G)|$.

Le degré $d(u)$ d'un sommet u est le nombre d'arêtes incidentes à ce sommet.

Nous noterons généralement m le nombre d'arêtes d'un graphe G.

Exercices de combinatoire:

Soit G un graphe d'ordre n. Quel est le degré maximum d'un sommet u dans ce graphe? Quel est le nombre maximal d'arêtes dans un tel graphe?

Bien évidemment, il y a un lien entre le nombre d'arêtes de G et l'ensemble des degrés des sommets de G; nous avons en effet le:

Théorème: Soit G un graphe d'ordre n ayant m arêtes. On note $V(G) = \{v_1, v_2, ..., v_n\}$.
Alors: $\sum_{i=1,2,...,n} d_G(v_i) = 2m$.

b- Chaîne, cycles et connexité.

Une *chaîne* (de longueur $l > 0$) dans le graphe G est une séquence d'arêtes $\mu = (e_1, e_2, ..., e_l)$ telle que pour $i \in \{1, 2, ..., l-1\}$, e_i et e_{i+1} aient une extrémité en commun. Ainsi, l'arête e_i a une extrémité en commun avec l'arête e_{i-1} et l'autre avec l'arête e_{i+1}.

Une chaîne μ est dite *élémentaire* si elle ne passe pas deux fois par le même sommet.

Une chaîne sera dite *simple* si elle ne passe pas deux fois par la même arête.

Un *cycle* est une chaîne simple dont l'arête initiale e_1 et l'arête finale e_l ont un sommet en commun.

Un graphe G est dit *connexe* si pour toute paire de sommets u, v de G, il existe dans G une chaîne dont les extrémités sont u et v (ie cette chaîne relie les sommets u et v).

Un sous-graphe connexe $[U]$ de G maximal pour l'inclusion sera dit être une composante connexe de G. L'ensemble des k composantes connexes d'un graphe G forment une partition de ses sommets en k-classes.

Exercice d'algorithmique: notion de concaténation

Est-ce qu'une chaîne élémentaire est simple? Une chaîne simple est-elle élémentaire?

A partir d'une chaîne $\mu = (e_1, e_2, e_3, ..., e_l)$ reliant les sommets u et v, il est aisé d'obtenir une chaîne élémentaire

reliant ses deux sommets: on définit l'opération suivante, dite de *concaténation*: soit x un sommet (différent de u et v) intervenant plus d'une fois dans la chaîne μ. On note e_i la première arête de μ ayant le sommet x pour extrémité, et e_j la dernière arête de la chaîne μ ayant x pour extrémité. Alors, la chaîne $\mu' = (e_1, e_2, ..., e_{i-1}, e_i, e_j, e_{j+1}, ..., e_l)$ est une chaîne reliant les sommets u et v et ne passant qu'une fois par le sommet x.
Comment adapter cette transformation de l'espace des chaînes afin de pouvoir éliminer des occurences multiples des sommets u et v?
Construisez un algorithme $\mathcal{A}$ permettant de transformer une chaîne quelconque μ reliant les sommets u et v en une chaîne élémentaire μ' reliant ces sommets.
En utilisant l'algorithme précédent, démontrez algorithmiquement la proposition suivante:
Pour toutes paires de sommets u, v du graphe connexe G, il existe au moins une chaîne élémentaire les reliant.
Notion de complexité algorithmique: Soit n l'ordre du graphe connexe G et l la longueur de la chaîne initiale. Estimez (grossièrement) la complexité de l'algorithme $\mathcal{A}$ produit en ces paramètres.
Paramètre du graphe: Composantes connexes, distance, diamètre.
On définit la classe de connexité d'un sommet u comme étant l'ensemble des sommets v tel qu'il existe une chaîne reliant u à v. (identique à la notion de composante connexe contenant u).
La distance $d(u, v)$ entre deux sommets u et v est définie comme étant la longueur de la plus petite chaîne reliant u à v dans G.
On définit le diamètre ($\delta(G)$) d'un graphe connexe G comme étant le maximum sur toutes les paires de sommets (u, v) de $V(G)$ de la distance $d(u, v)$, ie $\delta(G) = max_{u \neq v} d(u, v)$.
Exercice d'algèbre (théorie des relations).
On définit la relation $\mathcal{R}$ sur l'ensemble $V(G)$ par :
$u\mathcal{R}v \Leftrightarrow$ "$u = v$" ou "$u \neq v$ et il existe dans G une chaîne reliant u et v".
Démontrez que cette relation est une relation d'équivalence (réflexive, symétrique et transitive).
On peut donc partitionner l'ensemble $V(G)$ selon les classes d'équivalence de cette relation: ce sont les composantes connexes du graphe G.

c- Exemples de graphes: Le graphe social des élèves de l'IUT.
On considère le graphe (implicite) dont l'ensemble des sommets est l'ensemble des élèves de l'IUT, et dont les arêtes modélisent la relation d'amitié entre deux élèves, ie on met une arête $\{u, v\}$ si les élèves u et v sont amis.
Dans un tel graphe social, comment seront représentés les élèves particulièrement sociaux ou inversement les solitaires ?
On peut regarder le *degré moyen* des sommets du graphe: comment le définiriez-vous? Il permet donc de comparer une sociabilité moyenne avec le degré de chaque élève afin de savoir s'ils ont une sociabilité au-dessus de la moyenne ou non.
Quel pourrait-être le graphe modélisant une "bande de copains"? (les personnes se connaissent mutuellement).
Si l'on considère à présent uniquement l'ensemble des élèves de la filière informatique, que serait ce graphe par rapport au premier graphe ?
Si l'on fait les mêmes calculs et que l'on trouve que le degré moyen dans le sous-graphe G est bien plus haut que le degré moyen du graphe G, que peut-on en déduire?
Qu'est ce que la distance entre deux sommets u et v modélise ?
Que représente alors le diamêtre $\delta(G)$ du graphe G ?
A votre avis, si nous réalisions effectivement ces graphes pour les différents départements des universités toulousaines, quelles seraient les caractéristiques probables pour les différents paramètres du graphe? ie: A combien pourrait-on évaluer le nombre de sommets? Quel serait le degré moyen des sommets? A combien pouvez-vous alors estimer le nombre d'arêtes de ce graphe? Ce graphe serait-il connexe? Si nous réunissons les différents élèves dans des classes

représentant leur cursus actuel, quelle serait l'allure du graphe?
Dans un tel graphe G, on peut chercher à partitionner l'ensemble de ses sommets $V(G)$ en classes telles que les degrés moyens des classes soient forts et qu'il y ait peu d'arêtes entre deux classes différentes: on reconstitue alors les "classes sociales" du graphe. Etonnamment, les structures de ces décompositions semblent se retrouver dans de nombreux problèmes pourtant a priori différents. On parle ainsi de graphes de petits mondes.

Chapitre 2: Codage d'un graphe et conséquences.

Nous avons défini dans le chapitre précédent la notion de graphe. Maintenant, comment la rendre utilisable algorithmiquement par un ordinateur? Quelle structure de données implémenter?

a- Matrice d'adjacence d'un graphe.

Etant donné un graphe $G = (V(G), E(G))$ d'ordre n, nous pouvons lui associer la matrice A de dimension $n \times n$ faisant apparaître la relation d'adjacence entre les sommets. L'ensemble des sommets G sera ici: $V(G) = \{1, 2, 3, .., n\}$. Notons ici qu'une matrice $n \times n$ peut être vue comme une relation sur un ensemble de n éléments.

Les coefficients de la matrice d'adjacence $A(G)$ associée au graphe G sont définis par: $a_{i,j} = 1$ si les sommets i et j sont voisins dans G, et $a_{i,j} = 0$ sinon.

Exercice

Démontrez que la matrice d'adjacence d'un graphe G est symétrique.

Démontrez que la somme des coefficients de la ligne i est égale au degré du sommet i.

Soit m le nombre d'arêtes du graphe G. Combien $A(G)$ comportera-t-elle de coefficients non nuls?

En fait, il est aisé de voir qu'il existe une bijection entre l'ensemble des matrices $n \times n$ à coefficients dans $\{0; 1\}$ et dont la diagonale est nulle et l'ensemble des graphes d'ordre n dont l'ensemble des sommets est $\{1, 2, 3, ..., n\}$. Combien existe-t-il alors de graphes "différents" d'ordre n?

b- Matrice d'incidence du graphe G.

Soit G un graphe dont l'ensemble des sommets est: $\{v_1, v_2, ..., v_n\}$, et dont l'ensemble des arêtes est $\{e_1, e_2, ..., e_m\}$. On défnit alors la matrice d'incidence $A_I(G)$ comme étant la matrice à n lignes (sommets) et m colonnes (arêtes) telle que le coefficient $a_{i,j} = 1$ si l'arête e_j est incidente au sommet v_i, et 0 sinon.

Exercice

Quelle est la somme des coefficients d'une colonne quelconque de la matrice $A_I(G)$?

A quoi correspond la somme des coefficients de la ligne i de $A_I(G)$?

L'intérêt de la matrice d'incidence comparativement à la matrice d'adjacence réside partiellement dans un gain de place dans le stockage de ces matrices dans la mémoire de l'ordinateur.

Considérons le graphe des synonymes de la langue française défini comme suit: L'ensemble des sommets est l'ensemble des mots de la langue française, et les sommets i et j sont voisins dans ce graphe s'ils sont synonymes.

A combien estimez-vous le nombre de sommets ?

A combien estimez-vous le degré moyen d'un sommet dans ce graphe? Déduisez alors une estimation du nombre d'arêtes dans G.

Si $A(G)$ désigne la matrice d'adjacence de G, combien a-t-elle de coefficients?

Si $A_I(G)$ est sa matrice d'incidence, combien a-t-elle de coefficients?

c- Notion d'isomorphisme

Problématique: Le problème implicitement soulevé par le codage d'un graphe est le suivant: étant donné un graphe G que l'on ne connaît que graphiquement (dessin avec des sommets et des arêtes sans autre indication), nous pouvons lui associer une matrice, et pour se faire, il faut déterminer un ordre sur l'ensemble de ses sommets. Or, à ce niveau, apparaît un choix arbitraire. Un même graphe peut ainsi être associé à deux matrices "différentes" selon l'ordre dans lequel on considère les sommets du graphe G. De même, si l'on donne ces deux matrices à une personne et qu'on lui demande de dessiner effectivement le graphe associé, le résultat sera a priori très différent alors même que les deux matrices codent le même graphe. Ces problèmes, très difficiles pour l'heure, sont les problèmes d'isomorphisme entre graphes.

Deux graphes G_1 et G_2 sont dits isomorphes s'il existe une application $\phi : V(G_1) \to V(G_2)$ qui conserve l'adjacence,

c'est-à-dire: $\{u, v\} \in E(G_1) \Leftrightarrow \{\phi(u); \phi(v)\} \in E(G_2)$

Exercice algèbre

Démontrez que la relation d'isomorphisme définie précédemment est une relation d'équivalence sur l'ensemble des graphes, ie qu'elle vérifie les propriétés suivantes:

Réflexive, donc que tout graphe G est isomorphe à lui-même.

Symétrique, donc que s'il existe ϕ entre $V(G_1)$ et $V(G_2)$ vérifiant les propriétés d'un isomorphisme, alors il existe une application $\psi : V(G_2) \rightarrow V(G_1)$ qui vérifie ces mêmes propriétés.

Transitive: si G_1 est isomorphe à G_2 et que G_2 est isomorphe à G_3, alors G_1 est isomorphe à G_3.

d- Dénombrement des chaînes de longueur k.

L'un des avantages d'utiliser un codage matriciel classique est que nous avons de bonnes connaissances sur les opérations sur les matrices, et que ces connaissances nous permettent d'effectuer des calculs bien plus faciles. En voici un exemple parmi d'autres (à venir..)

Théorème Le nombre de chaînes de longueur k entre deux sommets i et j de G est égal au coefficient $a_{i,j}^{(k)}$, coefficient qui est à l'intersection de la ligne i et de la colonne j de la puissance k^{ieme} de la matrice d'adjacence $A(G)$.

e- Exercice de complexité algorithmique.

Nous allons ici formaliser un peu plus la notion de complexité d'un algorithme. Ce passage est volontairement très simplifié, et ne présente qu'une possibilité de définir la complexité d'un algorithme, mais cela donnera une première idée des méthodes du domaine. Si $\mathcal{A}$ est un algorithme dont on veut estimer la complexité, on décompose $\mathcal{A}$ en opérations élémentaires, qui sont par exemple l'addition, la multiplication, la comparaison, l'affectation ... Mais ce nombre d'opérations élémentaires effectuées par l'algorithme est dépendant du type d'ordinateur sur lequel il est implémenté ainsi que du langage de programmation dans lequel il est programmé. C'est pourquoi on ne s'intéresse qu'au comportement asymptotique de l'algorithme, c'est-à-dire en ne tenant compte ni des constantes multiplicatives, ni des constantes additives. On obtient alors une mesure intrinsèque de la complexité de l'algorithme $\mathcal{A}$, qui est généralement fonction des paramètres classiques du graphe, comme l'ordre n ou le nombre d'arêtes m.

Soit A une matrice $n \times n$. Quelle est la complexité de la multiplication matricielle dans le cas des matrices carrées d'ordre n ?

Chapitre 3: Graphes eulériens et graphes hamiltoniens.

a- Graphes connexes. (Rappel)
Un graphe G est dit connexe si et seulement si deux sommets quelconques de G peuvent être joints par une chaîne.
Exercice d'algèbre: théorie des relations
Nous définissons la relation $\mathcal{R}$ sur l'ensemble $V(G)$ par: $u\mathcal{R}v \Leftrightarrow$ "$u = v$" ou $u \neq v$ et il existe dans G une chaîne reliant les sommets u et v.
Démontrez que la relation $\mathcal{R}$ est une relation d'équivalence sur l'ensemble $V(G)$ des sommets de G, donc qu'elle est réflexive, symétrique et transitive.
Nous pouvons donc partitionner l'ensemble $V(G)$ selon les classes d'équivalences de cette relation: ce sont alors les composantes connexes du graphe G.
Algorithme simple de connexité
Nous allons décrire ici un algorithme simple donnant les classes de connexités d'un graphe G. Chaque sommet $x \in V(G)$ dispose d'une marque $M(X) = (i, t)$, avec i un entier et t un booléen.
$t = 0$ à l'initial, et $t = 1$ signifie que le sommet correspondant aura été testé (la marque sera alors définitive). Lorsque $t = 1$, i désignera la classe de connexité du sommet.
Initialement, $M(x) = (0, 0)$ pour tous les sommets $x \in V(G)$, et $j := 1$.

(1) Tant qu'il existe au moins un sommet x tel que $M(x) = (0, 0)$ faire:
(2) Choisir x tel que $M(x) = (0, 0)$.
(3) $M(x) := (j, 1)$.
(4) Tous les voisins v de x prennent la marque $M(v) = (j, 0)$
(5) Tant qu'il existe un sommet u avec $M(u) = (j, 0)$ faire:
(6) $M(u) := (j, 1)$.
(7) Tous les voisins v de u prennent la marque $M(v) = (j, 0)$.
(8) $j := j + 1$.

Exercice de complexité algorithmique.
Déterminez la complexité de l'algorithme précédent.

b- Graphes eulériens.
Soit G un graphe. Un cycle passant une fois et une seule par chaque arête de G est dit être un *cycle eulérien*. Un graphe sera dit *eulérien* s'il dispose d'un cycle eulérien.
Modélisation
Soit un graphe dessiné sur du papier (une enveloppe par exemple). Peut-on le dessiner sans lever le crayon du papier, en ne repassant jamais deux fois sur une même arête, mais en pouvant repasser plusieurs fois sur un même sommet ?
Plus historique: peut-on traverser les 7 ponts de Konigsberg et revenir chez soi par un bel après-midi de dimanche sans jamais emprunter deux fois le même pont?
Théorème:(Euler, 1736):
Un graphe connexe non trivial G admet un cycle eulérien si et seulement si chaque sommet de G est de degré pair.
Exercice d'algorithmique: Estimez la complexité du calcul des degrés des sommets d'un graphe G.

Méthode d'obtention pratique d'un cycle eulérien.
Soit G un graphe connexe. Un *isthme* est une arête dont la suppression augmente le nombre de composantes connexes

du graphe.
Importance de la notion d'isthme pour l'obtention d'un cycle eulérien:
Si G est un graphe disposant d'un isthme $\{ab\}$ et que l'on se trouve au sommet a, peut-on emprunter cet isthme et finalement obtenir un cycle eulérien du graphe G ?
L'algorithme suivant permet de construire, s'il existe, un cycle eulérien d'un graphe connexe:
Algorithme de Fleury.
a- On part d'un sommet a quelconque et l'on suit une arête quelconque.
b- Arrivé à un sommet $v \neq a$ à la k^{ieme} étape, on ne prend jamais une arête qui, au moment considéré, est un isthme pour le graphe G privé des arêtes déjà utilisées, sauf si c'est le seul choix possible.
c- Si on arrive au sommet a, on repart par une arête quelconque (non isthme si possible) si elle existe; sans cela, on s'arrête et on a parcouru un cycle eulérien.

Graphes hamiltoniens.
Soit G un graphe. Un cycle de G sera dit *hamiltonien* s'il passe une fois et une seule par chacun des sommets de G (sans emprunter nécessairement toutes les arêtes de G). Un graphe disposant d'un cycle hamiltonien sera dit être un *graphe hamiltonien*.
Modélisation.
Le responsable d'un site archéologique organise des visites de ses fouilles. Il dispose d'un plan, sur lequel il indique les lieux de fouilles (des sommets), et des chemins entre ces lieux de fouilles (les arêtes) qui ne présentent pas d'intérêt archéologique. Il souhaite trouver un cycle amenant ses futurs visiteurs une fois et une seule devant chacun des sites de fouilles. Est-ce possible?

Contrairement aux graphes eulériens, pour lesquels nous avions une caractérisation exacte et simple (degré des sommets), nous n'en connaissons pas actuellement de générale et utilisable pour les graphes hamiltoniens. Nous allons présenter un exemple de résultat pour les graphes hamiltoniens.
Théorème Un graphe G d'ordre n tel que pour tout sommet x, $d_G(x) \geq n/2$ admet un circuit hamiltonien.
Le théorème suivant affine celui-ci:
Théorème de Ore, 1961: Un graphe G d'ordre $n \geq 3$ tel que, pour toute paire $\{u, v\}$ de sommets non adjacents, $d_G(u) + d_G(v) \geq n$ admet un cycle hamiltonien.

Décomposition en cycle hamiltonien.
Nous pouvons aussi nous demander s'il est possible de décomposer un graphe en union de cycles hamiltoniens.
Soit G un graphe. Le graphe G est dit être décomposable en sous-graphes $H_1, H_2, ..., H_k$ si H_i et H_j n'ont jamais deux arêtes en commun, et si l'union des H_i $(1 \leq i \leq k)$ forme le graphe G. Nous pouvons alors nous intérésser à la décomposition de G en cycle hamiltonien.
Modélisation
Considérons par exemple que le site archéologique contienne 7 lieux de fouilles à visiter, et que tous les chemins soient possibles entre ces différents lieux. L'affluence risquant d'être au rendez-vous, le responsable souhaite ouvrir le plus grand nombre possible de voies différentes pour visiter son site archéologique, mais les voies ne doivent se croiser qu'aux 7 lieux de fouilles, et un chemin entre 2 lieux de fouilles ne doit appartenir qu'à une seule voie. Combien au maximum peut-on ouvrir de voies différentes ?
Théorème de Lucas: Le graphe complet K_{2n+1} admet une décomposition en n cycles hamiltoniens.

Chapitre 4: Coloration d'un graphe et initiation à la complexité.

a- Introduction: coloration.

Modélisation: On considère un ensemble de produits chimiques qu'il nous faut entreposer dans un minimum de salles, mais sous des conditions relatives à la sécurité des personnes et des locaux. Ainsi, certains produits chimiques ne doivent-ils pas être stockés dans une même salle, afin qu'en cas d'accident de maintenance, ils ne se mélangent pas. Nous modélisons donc cette situation à l'aide d'un graphe défini comme suit: Soit l'ensemble des sommets = l'ensemble des produits chimiques, et l'arête $\{i; j\}$ correspondra au fait que les produits i et j ne peuvent être entreposés dans la même salle pour raison de sécurité.

Nous définissons alors une coloration du graphe G ainsi obtenue comme étant une application c de $V(G)$ dans un ensemble de couleurs $\{1, 2, 3, ..., k\}$ vérifiant: $[c(i) = c(j)] \Rightarrow [\{i, j\} \notin E(G)]$, c'est-à-dire que deux produits peuvent être dans la même salle (couleur) si et seulement si ils ne sont pas incompatibles (donc ssi il n'existe pas d'arête entre les deux sommets dans G).

Nous voyons qu'il existe toujours une solution triviale à un tel problème d'affectation, à savoir disposer d'autant de couleurs que de sommets (on entrepose chaque produit chimique dans une pièce différente). Le problème de coloration d'un graphe G donné consiste à trouver le nombre minimum de couleurs $\chi(G)$ tel que le graphe G admette une $\chi(G)$ coloration.

Exemple Soit K_n le graphe complet ayant n sommets (et toutes ses arêtes). Que vaut $\chi(K_n)$?

Si K_{n_1,n_2} désigne le graphe bipartie complet, que vaut $\chi(K_{n_1,n_2})$.

Soit C_n un cycle de longueur n. Quel est le degré des sommets de C_n ? Combien C_n a-t-elle d'arêtes ? Inférez une hypothèse sur $\chi(C_n)$, et démontrez-la!

b- Introduction: Complexité d'un algorithme.

Le problème précédent de coloration relève de l'optimisation combinatoire, c'est-à-dire que l'on souhaite optimiser un critère (ici le nombre de couleurs que l'on souhaite minimiser) se traduisant en terme de problème combinatoire (à savoir dans notre exemple, étant donné un entier k, existe-t-il une k-coloration du graphe G, la réponse à cette question faisant intervenir de la combinatoire par le biais du nombre de tentatives qu'il faut faire afin de répondre oui ou non).

La résolution de certains de ces problèmes, lorsqu'elle est possible, se fait généralement au moyen d'un algorithme qu'il faut implémenter. Cependant, reste à savoir si notre algorithme est "réaliste" (c'est-à-dire se terminera avant quelques milliards d'années ou non). C'est pourquoi nous parlons aussi dans ce sours de la complexité d'un algorithme; il existe de nombreuses façons de mesurer la complexité d'un algorithme. Nous considérerons ici les bases simplifiées de cette discipline, en décomptant le nombre d'opérations élémentaires (addition, multiplication, comparaison...) faites par l'algorithme. On exprimera le résultat de cette complexité sous forme asymptotique afin de ne pas prendre en compte des paramètres non intrinsèques à l'algorithme tels le langage de programmation, la nature de l'ordinateur ... qui ajoutent des constantes additives.

En théorie des graphes, les paramètres usuels utilisés pour l'expression de cette complexité sont généralement n le nombre de sommets et m le nombre d'arêtes de l'algorithme.

Exemple: Complexité de la multiplication matricielle de deux matrices carrées $n * n$: $O(n^3)$.

A titre indicatif, nous venons de considérer qu'une multiplication de deux nombres quelconques ne prend qu'une opération, quelques soient les nombres. Or il semble bien évident que de grands nombres sont plus difficiles à multiplier que de petits nombres. On peut ainsi faire intervenir la longueur du nombre dans son codage...

Soit M un entier. Tout ordinateur codant les nombres en base 2, quelle est la longueur de la chaîne de caractères codant M en base 2 ?

Si M et N sont deux nombres, quel est alors le nombre d'opérations élémentaires sur les bits que l'ordinateur effectue

pour additionner N et M en utilisant leurs expressions en base 2 ? (on omettra toute retenue dans le calcul de la complexité).
Ainsi, vous voyez qu'intervient le $log_2(Max(M,N))$ dans l'expression de la complexité des opérations élémentaires. Néanmoins, on omet aujourd'hui de plus en plus l'incidence de ces paramètres étant donnée la puissance des ordinateurs actuels, sauf dans certains cas précis où l'on souhaite affiner au maximum les temps de calculs...

c- Quelques résultats théoriques sur le nombre chromatique d'un graphe G.
Nous allons essenitellement voir deux fois le même résultat, mais le second sera plus fin que le premier!
Théorème de Brooks [1941]: Soit G un graphe, et on note par $\Delta(G)$ son plus haut degré.
Nous avons alors: $\chi(G) \leq \Delta(G) + 1$.
Exercice: Démontrez cette formule par récurrence.
Théorème de Brooks II Avec les mêmes notations, nous avons: $\chi(G) = \Delta(G) + 1$ si et seulement si:
• Soit $\Delta(G) \neq 2$ et G admet pour sous-graphe $K_{\Delta(G)+1}$.
• Soit $\Delta(G) = 2$ et G a un cycle de de longueur impaire pour composante connexe.

d- Exemple d'algorithme approchant le problème de coloration d'un graphe.
Nous allons considérer (cf amphi) un algorithme garantissant une solution approchant la solution optimale.
Nous verrons en particulier comment, mathématiquement, nous pouvons garantir qu'un tel algorithme approche la solution optimale d'un rapport de l'ordre d'au plus $3n/log_k(n)$.

e- Lien entre la coloration d'un graphe et le problème de coloration des cartes géographiques.
De même, nous allons voir comment nous pouvons modéliser une condition de frontières sur des cartes géographiques à l'aide de la théorie des graphes. Pour les explications: voir en amphi ou en TD !

Chapitre 5: Graphes orientés.

a- Introduction.
Nous avons jusqu'à présent considéré des graphes non orientés. Nous rappelons à ce propos que les arêtes de tels graphes sont des ensembles non-ordonnés de deux éléments $\{x, y\}$. Ces arêtes modélisaient une relation symétrique, telle par exemple le fait que deux mots soient synonymes, l'incompatibilité chimique entre deux produits ou encore des routes à double sens. Mais nous pouvons aussi considérer des relations non symétriques, telles l'implication, la domination, ou plus simplement des routes à sens unique. Le but de ce chapitre est de voir comment se modélisent de telles relations, et comment les théorèmes vus dans les chapitres précédents se transposent à cette nouvelle notion.

b- Définitions.
Soit V un ensemble d'éléments. Un *couple* d'éléments de V est une paire ordonnée (x, y) d'éléments de V. Si V désigne les sommets d'un graphe, $x, y \in V$, le couple (x, y) sera dit être un *arc*.
x sera l'extrémité initiale (ou l'origine) de l'arc (x, y), alors que y sera dit l'extrémité terminale de l'arc.
Un *graphe orienté* $\overrightarrow{G}$ sera la donnée d'un ensemble de sommets $V(\overrightarrow{G})$ et d'un ensemble d'arcs $A(\overrightarrow{G})$ entre ces sommets.
Vocabulaire. Soit x un sommet d'un graphe orienté $\overrightarrow{G}$. Contrairement au cas des graphes non orientés où l'on parlait seulement de "voisins" du sommet x, il nous faut à présent distinguer la notion de prédécesseur et de successeur. Pour un arc (x, y), y sera dit être le successeur du sommet x alors que x sera dit être le prédécesseur du sommet y. Enfin, l'arc $a = (x, y)$ sera dit "sortant" du sommet x et entrant au sommet y. Les sommets x et y étant respectivement les extrémités initiale et terminale de l'arc (x, y).
Nous noterons $\Gamma^+(x)$ l'ensemble des successeurs du sommet x, et $d^+(x)$ le cardinal de cet ensemble, et $\Gamma^-(x)$ l'ensemble des prédécèsseurs du sommet x.
Remarque: Nous ne parlerons pas ici de graphes mixtes, possédant à la fois des arcs et des arêtes. Ainsi un graphe orienté ne possèdera-t-il que des arcs, et aucune arête.
Théorème fondamental dans le cas orienté. Soit $\overrightarrow{G}$ un graphe orienté d'ordre n et ayant m arcs. Notons $V(\overrightarrow{G}) = \{x_1, x_2, ..., x_n\}$ l'ensemble de ses sommets. Alors: $\sum_{i=1}^{n} d^+(x_i) = \sum_{i=1}^{n} d^-(x_i) = m$.

c- Codage d'un graphe orienté.
Matrice d'adjacence. La matrice d'adjacence A associée au graphe orienté $\overrightarrow{G}$ d'ordre n est la matrice carrée $n \times n$ définie par: $a_{i,j} = 1$ si j est successeur de i, 0 sinon.
Alors que la matrice d'adjacence d'un graphe non orienté était symétrique, celle d'un graphe orienté ne l'est pas.
Exercice de combinatoire. Si on ne considère que des graphes orientés antisymétriques, c'est-à-dire que si l'arc (x, y) existe, l'arc (y, x) n'existe pas, combien existe-t-il de graphes orientés antisymétriques d'ordre n ?
Si l'on considère des graphes orientés quelconques, pouvant posséder à la fois l'arc (x, y) et l'arc (y, x), combien existe-t-il de tels graphes orientés d'ordre n ?
Matrice d'incidence
La matrice d'incidence du graphe $\overrightarrow{G}$ ayant n sommets et m arêtes est définie comme étant la matrice A_I de taille $n \times m$ telle que: $a_{i,j} = 1$ si le sommet v_i est l'origine de l'arc e_j, -1 si le sommet v_i est l'extrémité terminale de l'arc e_j et 0 sinon.

d- Chemin, circuit et forte connexité.
Définitions: Les chemins et circuits sont les notions équivalentes aux notions de chaînes et cycles qui n'étaient définis que dans le cas des graphes non-orientés. Dans le cas non orienté, une chaîne reliait deux sommets de façon symétrique (c'est-à-dire qu'une chaîne entre x et y était aussi une chaîne de y à x). Maintenant, un *chemin* étant

orienté, il peut exister un chemin de x à y dans un graphe orienté $\overrightarrow{G}$ sans qu'il existe réciproquement de chemin partant de y et allant à x.
Nous définissons aussi les *chemins simples* comme étant les chemins ne passant pas deux fois par le même arc. Les *chemins élémentaires* sont ceux ne passant pas deux fois par le même sommet du graphe orienté $\overrightarrow{G}$. Un chemin dont les sommets extrémités x et y seraient identiques est dit être un *circuit*.
Forte connexité. Nous allons définir une notion algébriquement similaire à la notion de connexité dans le cas des graphes orientés.
Un graphe orienté $\overrightarrow{G}$ est dit être *fortement connexe* si, pour tout coupe de sommets (x, y), il existe un chemin allant de x à y et un autre allant de y à x.
Exercice d'algèbre (théorie des relations) Soit $\mathcal{R}$ la relation définie sur l'ensemble des sommets d'un graphe orienté $\overrightarrow{G}$ par: $x\mathcal{R}y$ si $x = y$ ou s'il existe un chemin allant de x à y et un autre allant de y à x.
Montrer que $\mathcal{R}$ est une relation d'équivalence.

e- Graphe orienté eulérien et hamiltonien.
Nous pouvons aussi définir pour les graphes orientés les notions de *circuit eulérien* comme étant un circuit passant par chaque sommet, et une fois uniquement par chaque arc de $\overrightarrow{G}$. Nous avons une caractérisation des graphes orientés eulériens très simples et proche de la caractérisation non-orientée.
Théorème d'Euler. Un graphe orienté $\overrightarrow{G}$ est eulérien si et seulement si $\overrightarrow{G}$ est connexe et $d^+(x) = d^-(x)$ pour tout sommet x de $\overrightarrow{G}$.
Nous pouvons aussi définir un *circuit hamiltonien* d'un graphe orienté $\overrightarrow{G}$ comme étant un circuit passant par tous les sommets une fois exactement. Tout comme le cas non orienté, trouver une caractérisation simple des graphes orientés admettant un tel circuit est impossible à l'heure actuelle (trouver un circuit hamiltonien dans un graphe orienté est un problème $\mathcal{NP} - complet$). Néanmoins, nous pouvons avoir des caractérisations simples pour des familles de graphes orientés. Suit ici la plus simple de ces caractérisations.
Un tournoi $T = (V, A)$ est un graphe orienté complet, c'est-à-dire que pour toute paire $\{x, y\}$ de sommets, soit $(x, y) \in A$, soit $(y, x) \in A$.
Théorème de Moon: Soit T un tournoi fortement connexe d'ordre $n \geq 3$. Pour tout sommet x et pour tout entier $k \in \{3, 4, 5, ..., n\}$, il existe dans T un circuit de longueur k passant par x. En particulier, un tournoi est hamiltonien si et seulement si il est fortement connexe.
Problème du voyageur de commerce (TSP). Etant l'un des problème les plus connus de Recherche Opérationnelle, nous en donnerons une présentation en anticipant le cours sur la recherche de chemins de longueurs optimales.

Chapitre 6: Cheminement dans un graphe orienté

a. Introduction à la problématique:
Nous allons ici considérer des graphes $\overrightarrow{G}$ orientés. Nous pourrons utiliser les différents algorithmes et notions sur un graphe non orienté G en nous rapportant au graphe orienté $\overrightarrow{G}$ ayant même ensemble de sommets et dont les arcs seront l'ensemble $A(\overrightarrow{G}) = \{(x,y)/\{x,y\} \in E(G)\}$.
Nous avons vu précédemment comment dénombrer les chaînes d'un graphe ou encore des définitions / propriétés de graphes relatives à l'existence de chaînes. Reste à savoir comment, étant donné deux sommets A et B d'un graphe $\overrightarrow{G}$ obtenir un chemin allant de A à B.
Problème du Batelier, du Loup, de la Chèvre et du Choux Ce problème se modélise par un graphe (cf. amphi) et la question de sa résolution revient à celui de trouver un chemin entre deux sommets de ce graphe.

b. Deux algorithmes essentiels
Il existe deux stratégies simples et complémentaires pour se déplacer dans un graphe, et qui permettent d'exhiber, s'il en existe, un chemin entre deux sommets donnés.
Recherche en profondeur(Trémeaux, 1882, Tarjan 1972). Aussi dit DFS (depth-first search).
Principe: On part d'un sommet v_0.

> (1) On avance le plus loin possible dans le graphe sans former de cycle. Lorsque l'on ne peut plus avancer (i.e.: sans alors former de cycle ou s'il n'existe pas d'arc sortant au sommet où l'on est) on retourne en arrière ("Backtracking") jusqu'au dernier sommet pour lequel il existe un arc sortant non emprunté.
> S'il existe un tel sommet, on retourne en (1).
> Sinon, l'algorithme prend fin.

Théorème: L'ensemble des sommets ainsi marqués d'un graphe non orienté à partir du sommet v_0 forme la composante connexe du sommet v_0. (Attention: ce n'est pas la composante *fortement* connexe!).
Complexité: Cet algorithme est de complexité linéaire en nombre d'arc, puisque l'on emprunte au plus une fois un arc donné. Le fait de faire du "Backtraking" est linéaire en nombre de sommets, et au final, pour un graphe connexe ($m \geq n-1$) : $O(n) + O(m) = O(m)$.
Application: Vérifier qu'une base de donnée contenant un arbre généalogique est bien acyclique.
Ou encore, permet de sortir d'un labyrinthe planaire dont l'entrée et la sortie sont toutes deux sur la face infinie du labyrinthe.

Recherche en largeur Dit aussi BFS (breadth-first search).
On part du sommet v_0. Chaque sommet va être finalement muni d'une marque $L(v)$ correspondant à sa profondeur dans le parcours de $\overrightarrow{G}$ à partir de v_0.
Principe: Le sommet v_0 est le seul sommet au niveau 0: $L(v_0) = 0$.

> Tous les voisins (successeurs) de v_0 sont mis au niveau 1, ie: si v tel que $(v_0, v) \in A(\overrightarrow{G}$, $L(v) = 1$.
> Tous les successeurs de sommets de niveau 1 sont de niveau 2 ... et ainsi de suite.

Théorème Si $L(x) = k > 0$, alors le plus petit chemin allant de v_0 à x est de longueur k.
Complexité: Chaque arc est parcouru une seule fois, donc la complexité est en $O(m)$.

c. Problème du plus court chemin d'un graphe orienté valué positivement.
Problématique: Considérons un graphe valué positivement dont les sommets représentent des villes et dont les arêtes

sont les routes entre ces différentes localités. Les valuations (positives) (l_{uv}) des arcs $((uv)$ représenteraient les temps de parcours des routes entre les différentes villes de ce graphe. Le problème est de savoir quel chemin emprunter pour se rendre de la ville A à la ville B en un minimum de temps...

Algorithme de Dijkstra, Moore 1959 Pour résoudre ce problème sur un graphe orienté $\overrightarrow{G}$ arc-valué positivement, on considère la marque $E(v)$ suivante pour chaque sommet v:

$E(v) =$ | Un marquage booléen (0 ou 1) s_v
Le nom du prédécessur dans le chemin
Une valeur réelle $L(v)$

On utilise alors l'algorithme suivant:

Initialisation: La marque de A est $E(A) = (1, \emptyset, 0)$, et pour tous les autres somemts: $E(v) = (0, \emptyset, \infty)$
$S_0 = \{v_0\}$ (ensemble des sommets dont la marque est définitive).
A l'étape $i+1$ faire:
(1) Pour tout sommet de $\overline{S_i}$ (ie: tel que $s_v = 0$), remplacer $L(v)$ par: $min_{u \in S_i \cap \Gamma^-(v)}\{L(v), L(u) + l_{uv}\}$.
(2) Pour tout sommet v dont l'étiquette $L(v)$ a été modifiée précédemment, donner au deuxième paramêtre de $E(v)$ la valeur u.
(3) Déterminer la valeur $min_{v \in \overline{S_i}}\{L(v)\}$, et sélectionner un sommet pour lequel ce minimum est atteint. On note u_{i+1} ce sommet, et on pose $s_{u_{i+1}} = 1$.
(4) $S_{i+1} = S_i \cup \{u_{i+1}\}$
(5) Si $i = n-1$, alors Fin, sinon $i \leftarrow i+1$ et retourner en (1).

Théorème: Lorsque l'algorithme de Dijkstra Moore stoppe, $L(v)$ est la longueur du plus petit chemin allant de A à v. Pour obtenir un tel chemin, il suffit de remonter les arcs en partant de v et en usant de la seconde marque.

c. Problème du plus court chemin d'un graphe orienté valué.

Cette fois-ci, le graphe est valué non nécessairement positivement !
Remarque: Le graphe ne doit pas comporter de circuit de "longueur" négative...

Algorithme de Bellman-Ford (1958) Nous allons munir les sommets du graphe d'une étiquette qui changera à chaque itération de l'algorithme. Soit k l'ordre d'itération, et v un sommet. Nous avons:

$E_k(v) =$ | le nom du sommet prédécessur dans le chemin
Une valeur réelle $L_k(v)$

On utilise alors l'algorithme suivant:

Initialisation $k = 0$ $E_0(A) = (\emptyset, 0)$; $\forall v \neq A, E_0(v) = (\emptyset, \infty)$.
(1) $k \leftarrow k+1$ Pour tous les sommets $v \neq A$ calculer $L_k(v) = Min_{u \in \Gamma^-(v) \cup \{v\}}\{L_{k-1}(u) + l_{uv}\}$.
(2) Pour tous les sommets v tels que $L_k(v) \neq L_{k-1}(v)$, donner au premier paramêtre de $E_k(v)$ la valeur du sommet u réalisant ce minimum.
(3) Si pour tout sommet v, $L_{k-1}(v) = L_k(v)$ Fin; sinon, retourner en (1).

Théorème: Si $\overrightarrow{G}$ ne contient pas de circuit absorbant de longueur négative, à la fin de l'algorithme précédent, la marque finale $L_k(v)$ est la longueur du plus court chemin entre A et v. Ce plus court chemin est obtenu en remontant de prédécésseur en prédécésseur, la première marque de $E_k(u_i)$.

d. Remarque sur le principe de sous-optimalité

Sans entrer dans le détail de la démonstration de ces algorithmes, nous pouvons nous demander pourquoi user d'algorithmes donnant le chemin optimal de A à tout sommet v alors que l'on ne recherche que le chemin optimal allant de A en B. C'est parce que l'on utilise le principe dit de sous-optimalité suivant:
Si $C_{x,y}$ est un chemin optimal entre x et y, alors pour tout sommet $z \in C_{x,y}$, le sous-chemin $C_{x,z}$ de $C_{x,y}$ est un chemin optimal allant de x à z.

Chapitre 7: Arbres et arborescences

a- Premières propriétés des arbres.
Un arbre est une notion non orientée.
Exemple: L'exemple typique d'arbre est l'arbre généalogique.
Définition: Arbre. Un arbre est un graphe G connexe sans cycle.
Remarque: Pour qu'un graphe n'ait pas de cycle, il faut globalement qu'il ait peu d'arêtes. Inversement, pour qu'un graphe soit connexe, il faut qu'il ait globalement beaucoup d'arêtes. Ainsi, la notion d'arbre est-elle au juste milieu de ces deux tendances inverses.
Cette formulation admet différentes formulations équivalentes. Par exemple:
•G arbre $\Leftrightarrow$ entre 2 sommets quelconques de G, il existe une chaîne et une seule.
•G arbre $\Leftrightarrow$ G connexe et $|V(G)| = |E(G)| + 1$.

b- Premières propriétés des arborescences.
Une *racine* est un sommet r d'un graphe orienté $\overrightarrow{G}$ tel pour tout sommet s de $\overrightarrow{G}$, il existe un chemin (orienté) de r à s. Une *feuille* est un sommet f de $\overrightarrow{G}$ n'admettant pas de successeur.
Définition: Une arborescence est un graphe orienté connexe $\overrightarrow{G}$ possédant une racine r et dont les autres sommets x vérifient $d^-(x) = 1$.
Exemple d'utilisation d'une arborescence: La notation polonaise inverse est une façon de noter les expressions mathématiques sans utiliser de parenthèses et de façon non ambigue. Il s'agit "simplement" d'un parcours en profondeur dans une arborescence binaire...

c- Arbres couvrants de poids minimum d'un graphe.
Problème: On souhaite relier différents villages via un réseau électrique, mais en investissant le moins de fonds possibles. Pour ce faire, des études préliminaires sont faites sur le terrain afin de savoir quels sont les coûts de constructions de ces lignes entre les différents villages. Disposant de ces études préliminaires, quels sont les travaux à entreprendre effectivement ?
On modélise donc la situation par un graphe valué. Les sommets du graphe sont les villes, et les arêtes du graphe sont les éventuelles lignes à construire. Ces arêtes sont valuées par le coût de construction de ces lignes. Le but est donc de sélectionner un arbre dans le graphe qui soit de poids minimum. (Précisez pourquoi).
Algorithme de Kruskal - 1956- On numérote les arêtes e_i selon un ordre croissant de leurs valuations (la valuation $v(e_i)$ de l'arête e_i vérifie: $v(e_i) \leq v(e_{i+1})$). On construit alors un arbre couvrant du graphe G en considérant les arêtes e_i dans l'ordre de numérotation, et en retenant e_i si elle ne forme pas de cycle avec les arêtes préalablement selectionnées.
La complexité de cet algorithme est en $O(m^2)$.

Algorithme de Prim -1957-
Soit $V(G) = \{v_1, v_2, ..., v_n\}$ les sommets du graphe G. On obtient un arbre couvrant de G en utilisant l'algorithme suivant:
- A_1 l'arbre initial est réduit à un sommet quelconque.
- A chaque itération, on construit A_{p+1} en augmentant A_p de l'extrémité non marquée de l'une des arêtes de poids minimum dont exactement une extrémité est dans A_p.
Complexité en $O(n^2)$.

Chapitre 8: Problèmes de flot dans un réseau de transports.

a- Présentation de la problèmatique:
Une usine comporte généralement un gros réseau de canalisations, transportant l'eau d'une source unique (le point d'arrivée d'eau) vers une sortie unique: le tout à l'égout. Lors d'une extension de l'usine, on raccorde les anciennes canalisations aux nouvelles, augmentant donc les débits de certaines canalisations, jusqu'á saturation de certaines d'entre elles. On souhaite alors remplacer certaine de ces canalisation afin d'augmenter le flot global de l'usine.
Le but du problème est donc de savoir quelles sont les canalisations saturées qu'il serait donc souhaitable de remplacer par de nouvelles canalisations plus importantes.

Modélisation: Nous avons deux choses différentes (mais pourtant semblables) à modéliser:
- le réseau physique des canalisations de l'usine, avec la capacité intrinsèque de la canalisation.
- la quantité d'eau passant effectivement dans une canalisation.
Nous allons modéliser le réseau physique des canalisations par un graphe dont les sommets représenteront les extrémités des canalisations, et les arêtes les canalisations elles-même. Nous allons valuer ce graphe par les capacités des canalisations; ainsi $c((x,y))$ correspondra à la capacité i.e. débit maximal pouvant traverser la canalisation (x,y). Nous voyons aisément que le graphe que nous construisons est orienté (l'eau s'écoulant dans un sens spécifique), et valué positivement.
Pour modéliser l'écoulement effectif de l'eau au travers des canalisations, nous superposons au graphe précédent un second, différent du premier par les valuations $f((x,y))$ qui vaudont ici le débit réel (flux) de l'eau à travers la canalisation (x,y). Remarquons que ce flux respecte la loi des noeuds, à savoir qu'en un sommet donné, la quantité d'eau qui arrive doit être égale à la quantité qui en ressort.

b- Formalisation du problème:
Une *source* est un sommet d'un graphe orienté qui ne possède aucun arc rentrant ($d^-(S)=0$). A l'inverse, un *puits* est un sommet d'un graphe orienté qui ne possède aucun arc sortant ($d^+(P)=0$).
Un *réseau de transports* N est un graphe orienté contenant une source et un puits et tel que chaque arc $a=(u,v)$ soit valué par une capacité $c(a)\in N$.
Un *flot* dans un réseau de transports $\overrightarrow{G}$ est une fonction f définie sur l'ensemble des arcs de $\overrightarrow{G}$ vérifiant:
- Pour chaque arc $a\in A(\overrightarrow{G})$, $0\leq f(a)\leq c(a)$.
- Pour tout sommet $u\in V(\overrightarrow{G})$, $\sum_{v\in\Gamma^+(u)} f((u,v)) = \sum_{v\in\Gamma^-(u)} f((v,u))$.
Intuitivement, un flot représente la quantité d'eau parcourant le réseau de transports en une unité de temps. La première condition est juste la limitation due à la canalisation (elle ne peut transporter plus que sa capacité). La seconde condition est aussi dite Loi de Kirchhoff, et est la formulation de la loi des noeuds.

c- Algorithme de recherche du flot maximum dans un réseau de transports.
Nous allons ici utiliser un algorithme basé sur un marquage des arcs du graphe.
L'algorithme partira d'un flot initial f_0 identiquement nul, qui est toujours un flot admissible dans un problème de transports.
Le principe de cet algorithme est d'améliorer le flot courant en recherchant une "chaine augmentante" dans ce graphe (cf. amphi).

Algorithme (simplifié) de Ford et Fulkerson - 1956.

1. On part du flot initial f_0 identiquement nul.
2. Si f_k est le flot courant, on cherche une chaîne élémentaire entre s et t n'empruntant que des arcs a dont le flot $f(a)$ vérifie:
   $0 < f(a)$ si l'arc a est emprunté en sens contraire.
   $f(a) < c(a)$ si a est emprunté dans le "bon" sens.
3. S'il n'existe pas de telle chaîne augmentante, alors le flot est maximum - Fin.
   Sinon, ajouter 1 aux flots des arcs de la chaîne augmentante qui sont parcourus dans le bon sens et -1 aux autres arcs, puis incrémenter k et retourner en 2.

Quelques remarques:

- Nous pourrions augmenter de plus que 1 à chaque itération, afin d'aller beaucoup plus vite dans cet algorithme.
- La complexité de cet algorithme est de $O(m.n^2)$ dans le cas des réseaux de transports.
- Lorsque les capacités des arcs ne sont pas des entiers, on ne peut garantir la vitesse de convergence de l'algorithme.
- Il existe de nombreux problèmes dériés de celui-ci, par exemple le problème du flot maximum de coût minimum dans un réseau de transports dont les arêtes possèdent des coûts différents.