Livret des SOLUTIONS

Théorie des Graphes

michel.mainguenaud@insa-rouen.fr

1 Environnement des graphes

1.1 Bases de données

Select	distinct « multi-graphe »
From	Graphe
Group by	Origine, Destination
Having	count (*) > 1

Principe : partitionnement sur le couple (Origine, Destination) et puis compter le nombre de tuple (ici d'arcs) et doit être strictement supérieur à 1 pour être dans le cas d'un multi-graphe. Distinct car il n'y a pas d'élimination des doubles.

1.2 Algorithmique

Principe : Parcourir le graphe à partir du point a dans le sens direct (i.e., en suivant les flèches des arcs puisque nous sommes ici dans le cadre orienté) et de créer un ensemble des nœuds parcourus. La même chose est effectuée dans le sens indirect (i.e., en suivant les flèches en sens inverse) et de créer un deuxième ensemble de nœuds parcourus. Le premier ensemble contient les nœuds accessibles à partir de a et le deuxième contient l'ensemble des nœuds permettant d'accéder à a. L'intersection de ces deux ensembles permet d'avoir les nœuds qui sont accessibles et permettent d'atteindre a. On obtient donc la composante connexe qui contient a.

Entrée :	G : Graphe a : Nœud	Sortie :	X' : {Noeud}

Début
    $X_1 \leftarrow \{a\}$
    Pour tout $x \in X$ faire examiné[x] $\leftarrow$ faux ;

    Tant que $\exists\, x \in X_1 \wedge$ examiné[x] = faux faire
        Examine [x] $\leftarrow$ vrai
        Pour tout u : $(x, y) \in U \wedge y \notin X_1$
            $X_1 \leftarrow X_1 \cup \{y\}$
        Finpour
    Fintantque
    $X_2 \leftarrow \{a\}$
    Pour tout $x \in X$ faire examiné[x] $\leftarrow$ faux ;

    Tant que $\exists\, x \in X_2 \wedge$ examiné[x] = faux faire
        examine [x] $\leftarrow$ vrai
        Pour tout u : $(y, x) \in U \wedge y \notin X_2$
            $X_2 \leftarrow X_2 \cup \{y\}$
        Finpour
    Fintantque
    $X' \leftarrow X_1 \cap X_2$
Fin

Principe : On choisit au hasard un nœud et on détermine, grâce à l'algorithme précédent, la composante fortement connexe qui le contient. Ensuite parmi les nœuds qui ne font pas partie de la composante connexe, on prend au hasard un nouveau nœud et on détermine la composante fortement connexe qui le contient. On recommence jusqu'à ce que tous les nœuds appartiennent à une composante fortement connexe.

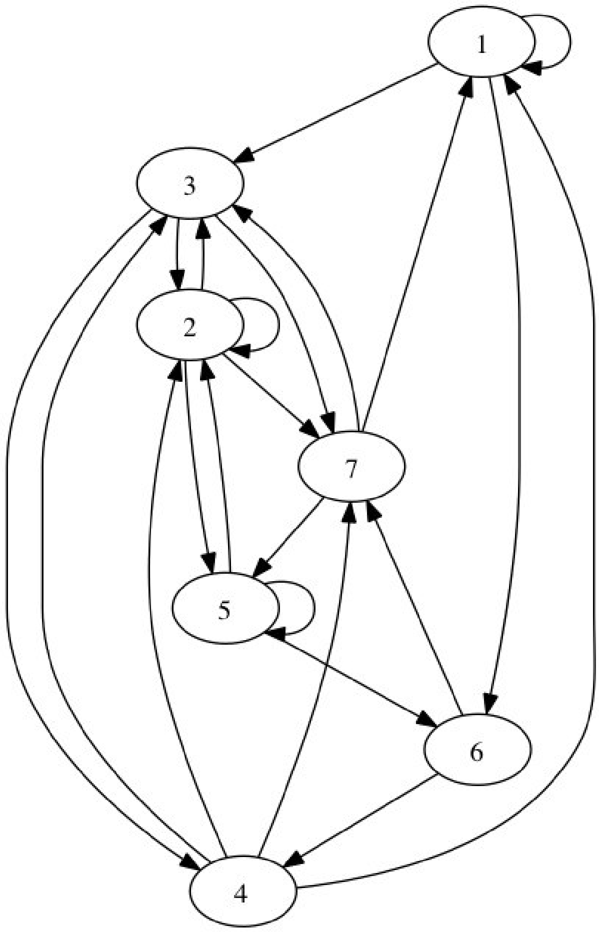
```
Entrées :       G : Graphe
Sortie :        C[1..N] : {Noeud}          // Nœuds de composantes fortement connexes
                i : entier
Début

        X' ← X
        i ← 1
        Tant que X' <> ∅ faire
                Choisir x ∈ X'
                C[i] ← ComposanteFortementConnexe (G, x)
                X' ← X' – C[i]
                i ← i + 1
        FinTantque
        i ← i - 1
Fin
```

2 Représentation physique :

2.1 Matrice Booléenne -> Graphe

1) Le graphe est orienté car la matrice est non symétrique (7, 5) vs. (5, 7)
2) Demi-degré extérieur : Γ+ (5) est le nombre de « 1 » sur la ligne 5 (ou le nombre d'arcs qui partent de 5). En l'occurrence ici : 3.
3) Demi-degré intérieur : Γ- (4) est le nombre de « 1 » sur la colonne 4 (ou le nombre d'arcs qui arrivent en 4). En l'occurrence ici : 2 (nombre de « 1 » dans la colonne 4).
4) Graphe : Il y a un arc entre deux sommets i et j si et seulement si il y a un « 1 » à l'intersection de la ligne i et de la colonne j



.

Représentation par graphe planaire : Quid d'un K3,3 ou d'un K5

Pas de K5 :	1	3,7,6,4	Pb : 6,3 => 1 n'existe pas
	2	3,7,5,4	Pb : 3,5 => 2 n'existe pas
	3	2,7,1,4	Pb : 2,1 => 3 n'existe pas

Pas de K3,3

2.2 Graphe -> Matrice Booléenne:

Réciproquement, l'intersection de la ligne i et la colonne j se voit affecter la valeur « 1 » si et seulement si il existe un arc du nœud i vers le nœud j. Autrement la valeur affectée est « 0 ». La matrice est carrée et contient autant de ligne que de nœud dans le graphe.
Attention ici le problème est de déterminer la matrice booléenne les seules valeurs acceptées sont donc « 0 » ou « 1 ». Le nœud A a deux arcs allant vers G. L'intersection de la ligne A et de la colonne G se voit affecter la valeur « 1 ». Il doit y avoir autant de « 1 » que d'arcs (exceptés les multi-arcs).

	A	B	C	D	E	F	G
A	0	1	1	1	0	0	1
B	0	0	1	0	1	1	0
C	0	1	0	1	0	0	1
D	0	0	0	1	0	1	1
E	0	0	1	1	1	0	1
F	1	0	0	0	0	1	1
G	0	0	0	0	0	1	0

$\omega(\{B, C, D\}) = \omega^+(\{B, C, D\}) \cup \omega^-(\{B, C, D\}) =$

$\{(B,F), (D,F), (D,G), (C,G), (B,E)\} \cup \{(A,B), (A,C), (A,D), (E,D), (E,C)\} =$

$\{(A,B), (A,C), (A,D), (B,E), (B,F), (C,G), (D,F), (D,G), (E,C), (E,D)\}$

Attention l'arc (B, C) par exemple ne compte pas car si l'origine appartient bien à {B, C, D} son extrémité (ici C) appartient aussi a {B, C, D}.

Cocycle extérieur : les 1 sur les lignes
Cocycle intérieur : les 1 sur les colonnes

3 Topologie

3.1 Chaîne/Chemin

Par exemple :

(1) Chaîne [u_1, u_2, u_3, u_4, u_5]
(2) Chaîne élémentaire [u_1, u_5] (ne passe pas 2 fois par le même nœud)
(3) Chemin (u_1, u_2, u_3, u_4, , u_5)
(4) Chemin élémentaire (u_1, u_5)

1.

3.2 Circuit

Par exemple :

(1) Circuit non élémentaire (u_1, u_2, u_3, u_4, u_5, u_6)
(2) Circuit élémentaire (u_1, u_2, u_6)
(3) Cycle [u_1, u_7, u_8, u_5, u_2, u_1] (peut passer deux fois par le même nœud)
(4) Cycle élémentaire [u_6, u_3, u_9, u_{10}]

3.3 Cocircuit / Cocyle

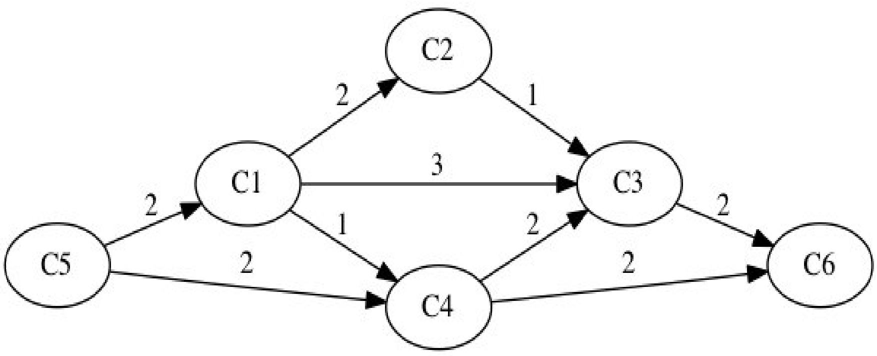
(1) cocircuit : ω+(W) = { (C, F) }, ω-(W) = { (B,C), (F, E) }
(2) cocycle : ω(W) = { [B, C], [C, F], [E, F] }

3.4 Connexité

Les composantes fortement connexes sont :

C1 = {A, B}, C2 = {C}, C3 = {D, E}, C4 = {F, G, H, I}, C5 = {R}, C6 = {S}

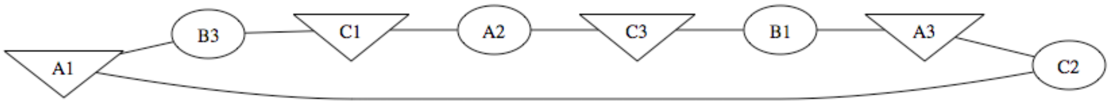
Le graphe réduit devient donc :



3.5 Echec et brillant

Les nœuds représentent les cases les arêtes (car non orienté : un mouvement ayant son inverse) les changements de cases.
La construction du graphe est faite a partir des mouvements possibles pour chaque case sachant qu'ici (b,2) ne pourra jamais être atteinte et ne permet d'aller nulle part, le graphe devient donc bi-parti. Le noeud est alors supprimé.
Le problème formel ici revient à calculer un chemin entre (a, 1) et [(a, 3) ou (c, 3)] sous contrainte que la case soit libre. Chacun des autres chevaux effectuant la même opération a chaque mouvement du cheval pris en référence. Ceci nous donne donc A1 - B3 - C1 - A2 - C3 pour un cheval ou dans l'autre sens pour sur le graphe pour rejoindre la destination (a, 3) ou (c, 3).

(A1 – B3) – (C1 – A2) – (C3 – B1) – (A3 – C2)
(B3 – C1) – (A2 – C3) – (B1 – A3) – (C2 – A1)
(A1 – B3) – (C1 – A2) – (C3 – B1) – (A3 – C2)
(B3 – C1) – (A2 – C3) – (B1 – A3) – (C2 – A1)



3.6 Tournoi

Les noeuds sont les concurrents, les arêtes sont les parties jouées entre les concurrents.
Chacun rencontre tout le monde => graphe complet.
La durée minimum est le nombre de couplage maximum possible / l'union des couplages donne le Kn.

K3 : nombre = 3 => 3 heures
K4 : 3 heures
K5 : 5 heures
K6 : 5 heures

graphe complet : n * (n - 1) /2 arêtes (6 * 5 /2) -> 15 arêtes 3 groupes par séances de jeu -> 5 parties : 5h
$T = |U| / |C_{max}|$

1: 1->2   3->4   5->6
2: 1->3   4->6   2->5
3: 1->4   2->6   3->5
4: 1->5   2->4   3->6
5: 1->6   2->3   4->5

$n = 5$ $C_{max} = 2$
$n = 6$ $C_{max} = 3$

3.7 Königsberg

1) Modélisation : Création d'un 2-graphe non orienté G (N, E), 4 noeuds N = {A,B,C,D}, 7 arêtes E = ([A,B], [B,C], [B,C], [A,C], [C,D], [C,D], [A,D]).

2) Théorème :

Condition nécessaire : Si G contient une chaîne eulérienne, tout noeud intermédiaire de cette chaîne est de degré 2, 4, 6, … suivant que la chaîne y passe 1, 2, 3, … fois. Si les noeuds extrêmes de la chaîne sont distincts, ils sont de degré, 1, 3, 5, … suivant que la chaîne y repasse 0, 1, 2, … fois. S'ils sont confondus, ils sont de degré pair : dans ce cas G admet un cycle eulérien.
Condition suffisante : Le théorème est vrai pour tout graphe à m-1 arête. Supposons le vrai pour tout graphe à (m-1) arête et soit G un graphe à m arêtes.
Cas no 1 : G a exactement 2 noeuds A et B de degré impair. Marquons les arêtes de G en partant de A en suivant une chaîne et nous nous astreignons à ne pas marquer 2 fois une même arête. Il est évident qu'une telle chaîne va se développer dans le graphe jusqu'à s'arrêter en un noeud de degré impair, donc en B. Soit L la chaîne obtenue :
Si L = E, elle est eulérienne
Si L <> E, soit G' le graphe partiel de G associé aux arêtes de (E-L), soient G1, …, Gp les composantes connexes de G'. Ces composantes ont moins de m arêtes et tous leurs noeuds sont de degré pair : elles contiennent donc chacune un cycle eulérien. D'autres part, chaque composante connexe possède au moins un noeud de la chaîne L, puisque G est connexe.
G contient alors une chaîne eulérienne. Pour l'obtenir, il faut partir de A et décrire la chaîne L. Chaque fois qu'on arrive à un noeud appartenant à un Gi, on ajoute à L le cycle eulérien associé à Gi et ainsi de suite.

Cas no2 : G n'a aucun sommet de degré impair. raisonnement identique. On part d'un noeud quelconque et L est ici un cycle.

3) Solution : Il n'y a pas de solution à ce problème en application même du théorème.

3.8 Degrés

$d(G) = ½ \Sigma (n \in X) d(n)$

$2 d(G) = \Sigma (n \in X) d(n)$

$\Sigma (n \in X) d(n)$ est paire

$d(1)+ \dots + d(n_k)$ est paire => le nombre de sommet de degré impair est 0 ou est pair.

Ou raisonnement par récurrence.

3.9 Graphe bi-parti

Un graphe bi-parti peut être décomposé en deux ensembles qui sont chacun des stables et donc les éléments ne sont pas adjacents entre eux. Ils peuvent donc être coloré de la même couleur. Il faut donc deux couleurs pour colorer un graphe bi-parti. Un cycle de longueur impaire (par exemple 3) nécessite au minimum 3 couleurs donc impossible.

3.10 Centre

Arbre de couverture minimale donc graphe connexe. Unique : non, il suffit de prendre un arbre restreint à deux noeuds.
Limité : oui, deux centres d'écartement n (C1, C2) sont forcément adjacents car sinon il existerait entre C1 et C2 un noeud. Le graphe est connexe. La longueur de C1 aux autres noeuds liés à C2 est donc n-1 car C1 est un centre. Si C1 et C2 ne sont pas adjacents la longueur des noeuds liés à C2 est de n-1 + la liaison de C2 au noeud intermédiaire + la liaison du noeud intermédiaire à C1 donc n+1 donc problème. Dans notre cas, nous avons affaire à un arbre, il ne peut donc pas y avoir plus de 2 noeuds adjacents deux à deux sinon il y a formation d'un cycle (impossible dans un arbre).

3.11 Eulérien

Pour qu'un graphe soit Eulérien, il faut et il suffit que tous ses sommets soient de degré pair.

Si un graphe contient k noeuds impairs, il est possible de rajouter un nouveau noeuds x, relié à ces k noeuds. Dans le graphe obtenu, les k noeuds considérés sont devenus pairs…
Cependant, le degré de x étant k, le graphe n'est toujours pas Eulérien si k est impair…

Ce n'est possible que si le nombre de noeuds impairs est pair…

RQ : il est possible de rajouter des arêtes entre les noeuds de degré impair dans le graphe d'origine… Mais l'ajout d'une telle arête, entre deux noeuds impairs a et b par exemple, fait que le nombre de noeuds impairs devient k-2, qui a la même parité que k…

3.12 Mon beau sapin

(a) Modélisation :
  - Noeud : machine
  - Non orienté car bi-directionnelle
  - Arête : connexion entre machine

(b) Détermination de la taille d'un graphe

(c) Un arbre d'ordre n possède n - 1 arêtes
1 forêt de k arbres : un arbre a_i a n_i noeuds
$\Sigma n_i = n$
Chaque arbre a_i a : $n_i - 1$ arêtes
Le nombre total d'arête est : n - k

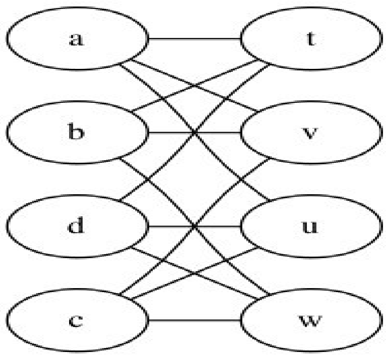
3.13 Bi-parti

a)
Nombre de nœuds : 11
Nombre d'arête : ordre = 11, d = 4 => taille = 11*4/2 = 22
Bi-parti :
=> les deux groupes sont des stables => pas d'arête entre les nœuds de chaque stable par définition

nombre d'arête = 4 x k1 (où k1 est l'ordre du premier bi-parti) => 4 x k1 = 22 => ordre non entier => impossible.

b)

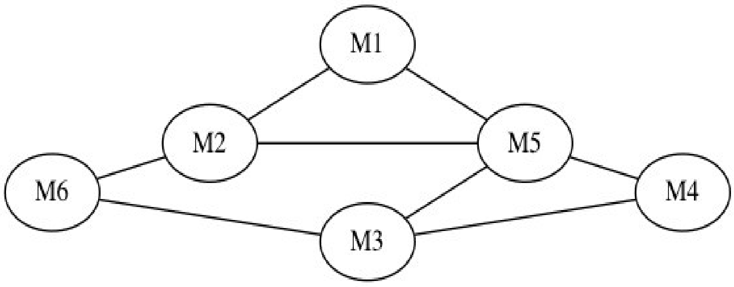
3 arêtes/noeud x k1 noeud = 12 arêtes => k1 = 4, k2 = 8 – 4 = 4 situation symétrique
d <= ordre => possible



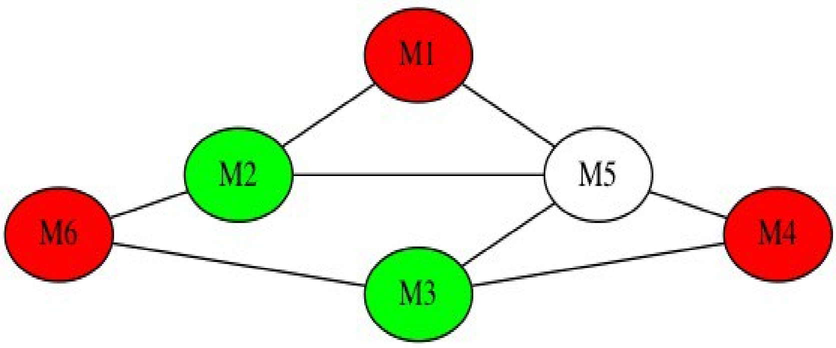
4 Propriétés

4.1 Examens

Il y a incompatibilité entre deux déroulements simultanés si un étudiant doit passer les deux matières. On va donc chercher un stable qui fait que deux matières ne sont pas connectées directement (elles peuvent donc se dérouler en même temps puisque aucun étudiant ne doit les passer toutes les deux). On construit un graphe où les noeuds sont les examens et une arête (pas besoin d'orientation ici) si un étudiant doit subir ces deux examens.



A partir de ce graphe, on cherche un ensemble stable : (M1, M6, M4). Le nombre maximal d'examen que l'on peut organiser par jour est donc la cardinalité du stable de cardinalité maximale (ici $\alpha = 3$).
Le nombre de jour correspond au nombre de couleur à utiliser pour colorier les noeuds de telle manière que deux noeuds adjacents ne soient pas de la même couleur (le nombre de couleur va déterminer le nombre de « composant » regroupant les matières qui peuvent avoir lieu en même temps (ici $\gamma = 3$).
L'intersection de 2 quelconques stables est nulle (les matières ne peuvent pas se dérouler simultanément) et l'union des stables forme l'ensemble initial des nœuds (on n'a oublié aucune matière).



4.2 Localisations potentielles

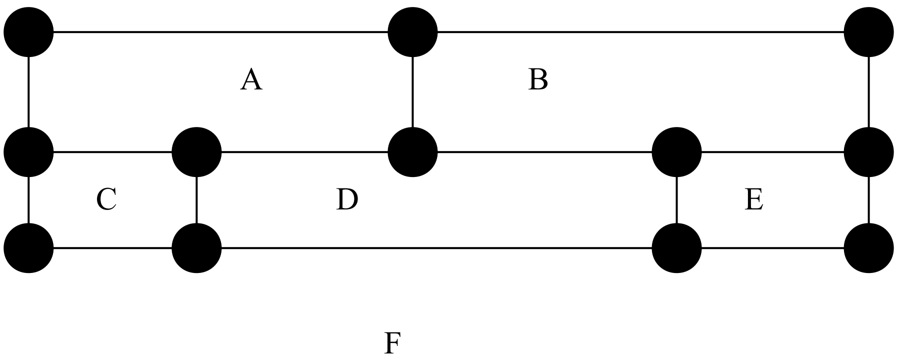
Les nœuds modélisent les installations potentielles. On définit une fonction d'étiquetage (ν) sur les nœuds permettant à chaque nœud de donner son bénéfice escompté. Les arêtes représentent les incompatibilités de localisation suite à une distance trop faible (deux nœuds sont reliés si leur distance entre les installations représentées est inférieure à 40 km). Le graphe est alors un graphe de la forme G(X, U, ν). Le problème se ramène donc à calculer un stable (deux nœuds ne doivent pas être adjacents) telle que la somme des bénéfices escomptés est maximale (agrégat sur la fonction d'étiquetage des nœuds).
L'alternative est de modéliser par les arcs une distance supérieure à 40 et le problème se ramène à la recherche d'un sous-graphe complet de poids maximal.

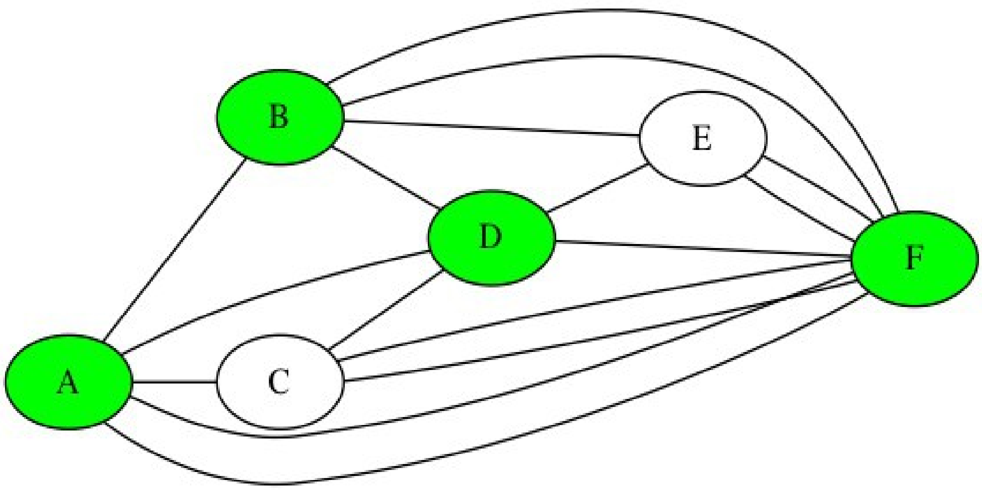
Graphe : Nœud : localisations potentielles
Arête : localisations ayant une distance inférieure à 40 Km.

Problème : Chercher le stable S = {li, …. lj} / Max $\sum_{i \in S} b(li)$

4.3 Crayon

On construit un graphe (ici un multi-graphe). Chaque noeud correspond à l'une des faces (avec une face extérieure). Les arêtes représentent les segments et relient deux noeuds associés aux deux faces adjacentes à ce segment.





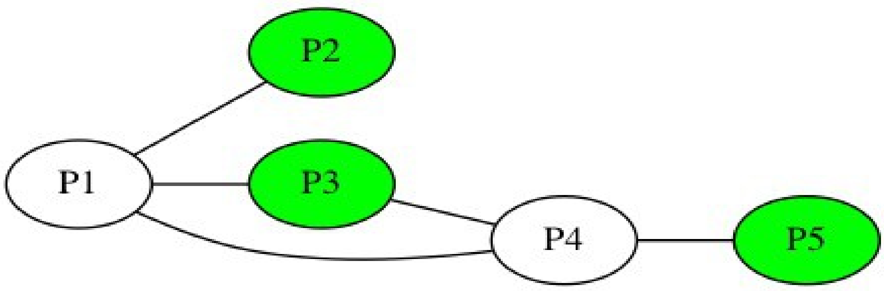
Le problème devient alors : comment définir une chaîne (il n'y a pas besoin de revenir au début) qui utilise une fois et une seule les arêtes (une fois pour ne couper qu'une fois le segment). Ceci se ramène donc à rechercher une chaîne eulérienne dans le graphe. A partir de la définition du théorème d'Euler il n'existe pas de chaîne eulérienne pour ce graphe. Un graphe admet une chaîne eulérienne si et seulement si le nombre de sommets de degré impair est 0 ou 2 (ici il y en a 4 : A, B, D et F).

4.4 Portes

Modélisation par graphe : Les nœuds vont modéliser les portes. Les arcs vont modéliser les incompatibilités d'ouverture. Le problème se ramène alors à rechercher des nœuds en cardinalité maximale qui ne sont pas adjacents. Le problème est alors de chercher un stable de cardinalité maximale.

Graphe : Nœud : correspond à une porte
Arête : correspond à une incompatibilité d'ouverture

L'objectif est de trouver les portes qui peuvent s'ouvrir simultanément et donc, celles qui sont ouvertes ne peuvent pas être adjacentes -> On recherche donc un stable de cardinalité maximale.



Stable de cardinalité maximale 3 : (P2, P3, P5)

Modélisation par programmation linéaire : Les portes représentent des variables dont le domaine est {0, 1}. Le problème est alors de maximiser la somme des variables portes sous les contraintes d'incompatibilités d'ouverture.
pi : porte valeur binaire (0 : fermée, 1 :ouverte)

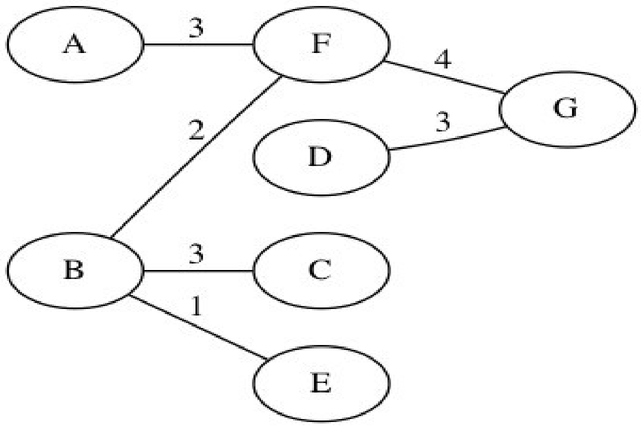
$$
\begin{aligned}
&\text{Max } p1 + p2 + p3 + p4 + p5 \\
&\quad p1 + p2 <= 1 \\
&\quad p1 + p4 + p3 <= 1 \\
&\quad p5 + p4 <= 1 \\
&\quad p1, p2, p3, p4, p5 >= 0 \\
&\quad p1, p2, p3, p4, p5 <= 1
\end{aligned}
$$

4.5 Arbre de valeur minimale

L'objectif consiste ici à créer un ensemble de sous arbres et de les relier entre eux avec les arêtes de moindre coût.

1) Choix de A : sélection de l'arête [A, F]
2) Choix de B : sélection de l'arête [B, E]
3) Choix de C : sélection de l'arête [C, B]
4) Choix de D : sélection de l'arête [D, G]

On se retrouve avec trois sous arbres (graphes disjoints par définition). Il faut maintenant relier ces sous arbres. Il existe des arêtes entre ces sous arbres. En ramenant chaque sous arbre à un noeud, on se retrouve avec un multi-graphe (plusieurs arêtes entre chaque représentant de sous arbre). Soient a = le sous arbre {A, F}, b = sous arbre {B, C, E}, c = sous arbre {D, G}.
Partant de a, l'arête la plus faible incidente extérieurement est [B, F]. On vient donc de relier a à b. L'arête la plus faible incidente extérieurement est [F, G]. On vient donc de relier (a,b) à c.
La valeur finale est : 16 = (3 + (3 + 1) + 3 + (2) + (4)). L'arbre de valeur minimale est (non unique) :



Le choix stratégique du positionnement des « epsilon » incombe au manager (aide à la décision et non système de décision). Exemple choix de B-C ou C-E de même coût (ici 3).

4.6 Fonction de Grundy

Le premier graphe ne contient pas de boucle, ni de circuit. Le deuxième graphe ne contient pas de boucle mais contient un circuit (A,D,C,B,A).
Pour le premier graphe : On ne peut affecter un nombre que lorsque tous ses successeurs ont eu un nombre qui leur a été affecté. Le graphe n'a pas de circuit on peut adopter un séquencement de type tri topologique et affecter le numéro à partir des noeuds n'ayant pas de successeurs ($\Gamma+(x) = \varnothing$).

L'attribution s'effectue donc sous la forme :

I, C = 0 ; B, D, H = 1 ; E = 0 ; G = 2 ; A = 2, F = 3

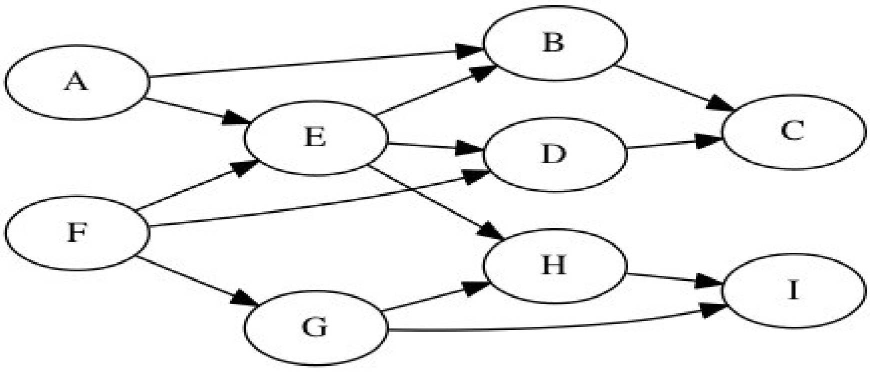
Formalisation : j et k sont des entiers

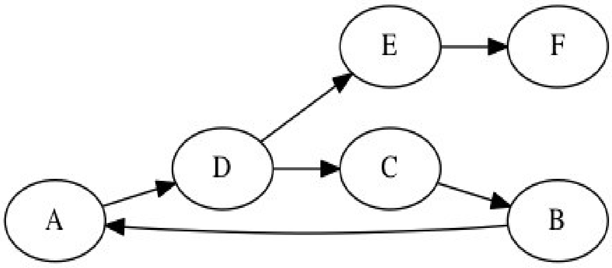
$g(x) = k > 0 \Rightarrow \forall\ 0 <= j < k\ \exists\ y \in \Gamma+(x)\ /\ g(y) = j$

$g(x) = k \Rightarrow \forall\ y \in \Gamma+(x)\ /\ g(y) \neq k$

Graphe sans circuit => fonction de Grundy => noyau (unique)

Cette formalisation définit la fonction de Gründy. Une fonction de Gründy n'existe pas toujours pour un graphe, de même qu'un graphe peut admettre plusieurs fonctions de Gründy.





Sur ce graphe deux fonctions sont possibles :

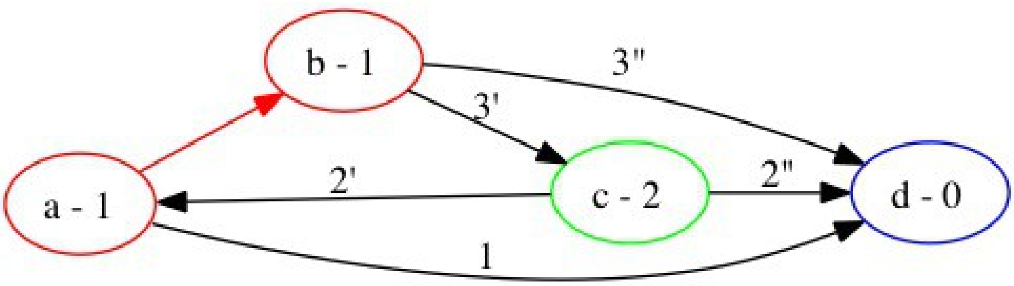
A, C = 0 ; B = 1 ; D = 2 ; E = 1 ; F = 0 ; ou B, D = 0 ; A, C = 1 ; E = 1 ; F = 0

Si un graphe admet une fonction de Gründy, il admet un noyau ($S = \{ x / x \in X, g(x) = 0\}$ car on a simultanément :

$x \in S \Rightarrow g(x) = 0 \Rightarrow \min (y \in \Gamma(x))\ g(y) > 0 \Rightarrow \Gamma(x) \cap S = \varnothing$

$x \notin S \Rightarrow g(x) > 0 \Rightarrow \min (y \in \Gamma\ (x))\ g(y) = 0 \Rightarrow \Gamma(x) \cap S \neq \varnothing$

L'existence d'un noyau n'implique pas l'existence d'une fonction de Grundy.
{d} est un stable absorbant et donc un noyau mais pas de fonction de Grundy.



Il y a un cycle, on casse le cycle (a,b) puis problème sur (a, b)
Pseudo-fonction de Gundy :

(1) $g(x) = k > 0 \Rightarrow \forall\ 0 <= j < k\ \exists\ y \in \Gamma+(x) / g(y) = j$

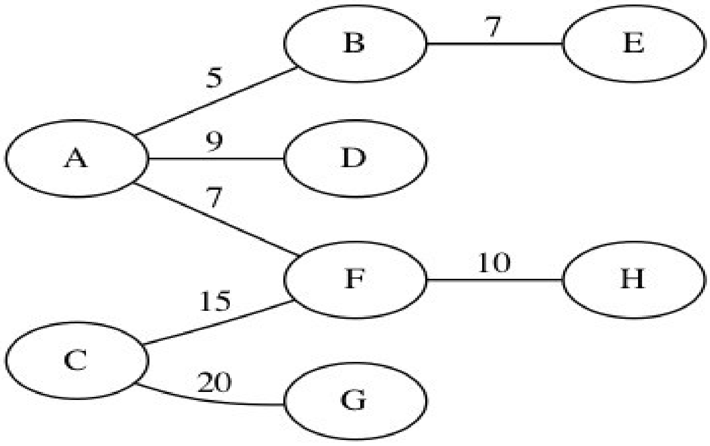
(2) $g(x) = 0 \Rightarrow \forall\ y \in \Gamma+(x) / g(y) \neq 0$

Il peut y avoir plusieurs nœuds avec la même valeur (non nulle) . Le noyau est {x / g(x)= 0}
Stable par (2), absorbant par (1).

4.7 Réseaux de communication

Noeud : machine, Graphe non orienté (symétrique), Arête : connexions
Problème : Graphe connexe sans redondance (cycle) → arbre

Le problème se ramène au calcul d'un arbre de recouvrement de valeur minimale (les étiquettes des arcs sont des coûts). L'application de l'algorithme de Sollin donne la valeur : 73



Alternative à l'algorithme de Sollin, l'algorithme de Kruskal (de type glouton) :

```
U ← Ø;
pour i ← 1 à m faire
        si    G (X, U ∪ {ui }) ne contient pas de cycle
        alors
                U ← U ∪ {ui }
        fsi
fpour
```

4.8 C'est la fête

Les noeuds : les délégués. Les arcs (i, j) quand le délégué i connait l'adresse du délégué j. Le graphe est orienté car la connaissance n'est pas symétrique. Pas de boucle.

« Tous les délégués seront-ils avertis » : Oui si quelle que soit la configuration des connaissances d'adresse, le graphe est fortement connexe sinon Non.

Soit un groupe de 8 délégués tel que chaque délégué connait l'adresse des 7 autres. Le sous-graphe associé est donc un K8. Il reste donc 7 autres délégués qui forment un sous-graphe fortement connexe plus une connexion vers le K8 (un délégué de ce sous-graphe : 6 connexions vers les autres pour former un groupe de 7 + une dernière connexion vers le K8).

Les connexions sont toujours du groupe de 7 vers le groupe de 8 donc le graphe est semi-fortement connexe mais pas fortement connexe. Si le délégué ASI4 appartient au K8 il ne pourra pas joindre un délégué du groupe de 7.

4.9 Arithmétique

Si n <= 20, on a n+1 <= 21, donc 1 est adjacent à tous les entiers <= 20. Ceci prouve que le graphe est connexe. Son diamètre est au plus 2 (on peut toujours trouver une chaîne de longueur au plus 2 de i à j en passant par 1 car 1 est adjacent à tous les noeuds). Comme (11, 12), (11, 13),...,(19, 20) ne sont pas adjacents, le diamètre est exactement de 2.

$\Gamma(1) = X - \{1\}$: connexe et $1 \in \Gamma(x)\ \forall x \in X - \{1\}$ => diamètre = 2

4.10 Nao

On applique la méthode matricielle pour déterminer la longueur du chemin (le plus court) entre chaque couple de nœuds. On prend la ligne qui minimise la somme de ces éléments (avec $a_{ii} = 0$).

4.11 Arithmétique le retour

Nombre premier : s'il est son seul diviseur avec 1. Chaque noeud est son diviseur donc chaque noeud a une boucle ($\nu(x_i)$ divise $\nu(x_i)$).

1 divise tous les $\nu(x_i)$ donc x_1 est relié à tous les noeuds.

On recherche donc tous les x_i / $\Gamma 2\ (x_i) = \{x_1, x_i\}$

Décomposition en facteurs premiers :

> Opérateur : $\rightarrow$ (A, B, longueur max) = $\{e_\kappa\}$
>
> Décomposition (X) : Π i = 1 à longueur ((1, X))$\varepsilon(e_\kappa)$ / $\varepsilon_\kappa \in \rightarrow (1, X)$

4.12 Arthur et les mini-chevaliers

Au lieu de passer par le graphe direct, on passe par le graphe complémentaire. L'objectif est alors de construire un circuit hamiltonien (retour au noeud de départ pour la table ronde) ou une chaine hamiltonienne (table en forme de U).

Le cycle/chaîne hamiltonien représente le plan de table.

5 Chemins

5.1 Evaluation de chemins

La séquence nœuds conduisant à la somme des valuations la plus faible est : (AHFDCJ) et sa valeur est : 12 ou (AHFGDCJ) par l'algorithme de Dijkstra.

5.2 Méthode matricielle

A	1	2	3	4
1	0	4	4	8
2	3	0	4	3
3	∞	∞	0	5
4	-6	∞	2	0

A^2	1	2	3	4
1	0	4	4	7
2	-3	0	4	3
3	-1	∞	0	5
4	-6	-2	-2	0

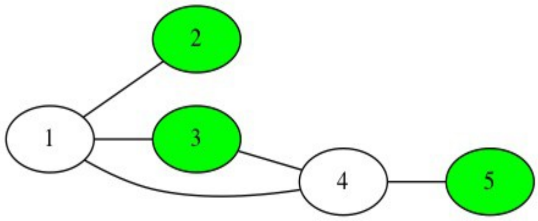
$A^3 = A^4$	1	2	3	4
1	0	4	4	7
2	-3	0	1	3
3	-1	3	0	5
4	-6	-2	-2	0

6 Parrallélisme

6.1 Wifi

Graphe : Nœud : Machine
Arête : incompatibilité de fonctionnement

Problème : Les machines retenues ne peuvent pas être adjacentes. On cherche donc des stables. Pour maximiser le nombre de machines on va chercher la cardinalité du stable de cardinalité maximale ici 3. (M2, M3, M5).



6.2 Migration de tâches

Problème de programmation linéaire :
On connaît la durée p_{ij} d'exécution de la tâche i sur la machine j. On recherche un ordonnancement de durée c_{max} minimale. Etablir l'ordonnancement c'est choisir la durée xij passée par la tâche i sur la machine j et le planning de chacune des machines.

Min c_{max}
// la somme des durées des morceaux de la tâche i est $\leq$ à c_{max}
Σ (x_{ij}, j=1, ..m) xij $\leq c_{max}$ (i = 1, .. n)
Σ (x_{ij}, i=1, ..n) xij $\leq c_{max}$ (j = 1, .. m) // la somme des durées sur la machine j est $\leq$ à c_{max}
Σ (x_{ij} / pij, j=1, … m) = 1 (i = 1, …, n) // la tâche i est entièrement exécutée
$x_{ij} \geq 0$ // valeurs positives

6.3 Wifi le retour

Graphe : Nœud : Machine
Arête : incompatibilité de fonctionnement

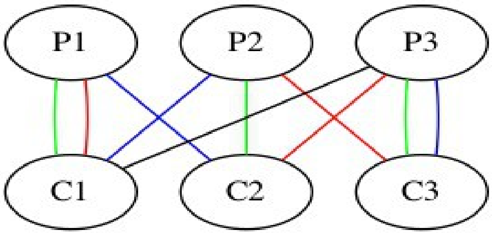
Problème : Les machines retenues ne peuvent pas être adjacentes. Les configurations retenues sont donc des stables. On recherche le nombre minimum de stables possibles dont l'union couvre toutes les machines (le nombre chromatique, ici 2).
(M2, M3, M4), (M1, M5)
(M2, M3, M5), (M1), (M4)

Cardinalité(Union(Si), i=1,..n), Si un stable de cardinalité maximum

6.4 Emploi du temps

Graphe bi-parti construit avec les nœuds sont les professeurs et les cours. Les arêtes (relation symétrique) sont les cours donnés par un professeur. Multi-Graphe.
Deux arêtes de même couleur ne peuvent pas être adjacentes => Indice chromatique

On obtient le graphe biparti coloré suivant :



=> 4 plages

	P1	P2	P3
Plage 1	C1	C3	C2
Plage 2	C1	C2	C3
Plage 3	C2	C1	C3
Plage 4			C1

6.5 Réseau

Noeuds : les ordinateurs, Arêtes : liaisons (non orienté car bi-directionnelle), Graphe simple (pas de boucle).

Problème : tracer un graphe régulier simple de degré 3.

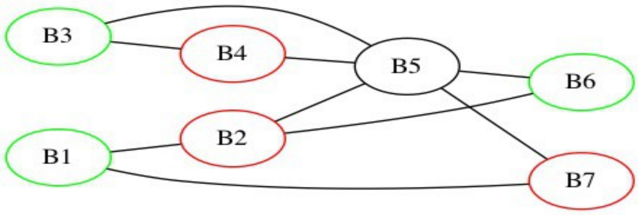
Si chaque appareil est relie a exactement 3 ordinateurs, les noeuds du graphe sont tous de degre impair. Le nombre de noeuds de degré impair est soit nul soit toujours pair, un tel graphe doit posse der un nombre pair de sommets, donc ici impossible.

6.6 Publication des bancs

Graphe :

Noeud : un banc
Arête : une allée (non orienté)

Deux bancs reliés directement ne peuvent pas être de la même couleur.

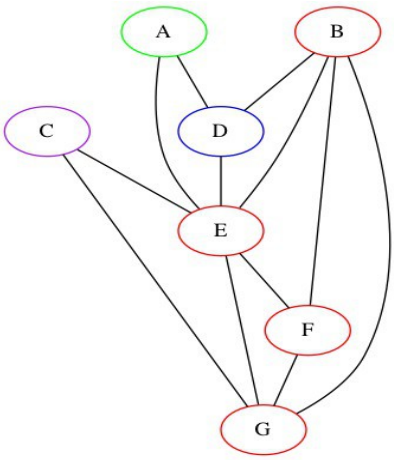


1. Nombre chromatique. Il faudra 3 couleurs.
2. Graphe connexe. Passez par toutes les allées une fois et une seule sans chercher à revenir au point de départ : chaîne eulérienne => nombre de sommet impair = 0 ou 2. Ici impair : B2, B5, => OK B5-B3-B4-B5-B6-B2-B5-B7-B1-B2 . En revenant au banc de départ : cycle eulérien => tous les nœuds de degré pair. Ici B5 impair => impossible.
3. Cycle hamiltonien ? Graphe hamiltonien Ici impossible (d(B3) + d(B7) = 4 < 7). La somme de toute paire de nœuds non adjacents vaut au moins |X| (le nombre de nœuds du graphe)

6.7 Après les bancs les sièges

Graphe :

Noeuds : les e le ves ; Arête (relation symétrique)(i, j) signale que les e le ves i et j se sont rencontre s.
Graphe simple.
Nombre de siège : Cardinalité de la clique (sous-graphe complet) maximale ici 4 (B, E, F, G)



7 Applications

7.1 Villes candidates

Graphe : Nœud : villes
Arête : liaison autoroutière

Les villes non retenues ne doivent pas être reliées entre elles directement. Cela signifie qu'elles forment un stable. L'ensemble des villes retenues est donc constitué du complémentaire de ce stable vis à vis de l'ensemble des villes. Elles forment donc un support.

7.2 Square-Dance

Graphe : Nœud : personnes
Arête : relie deux personnes qui peuvent danser ensemble

Ce problème se ramène a un problème d'affectation. Les couples sont formés par le choix d'une arête. Un nœud (personne) ne pourra pas danser avec deux personnes simultanément. Les arêtes retenues ne devront donc pas être adjacentes. On cherche donc un couplage de cardinalité maximum.

7.3 Cité-U

Tableau initial :

	a	b	c	d	e
A	1	2	3	4	5
B	1	4	2	5	3
C	3	2	1	5	4
D	1	2	3	5	4
E	2	1	4	3	5

On soustrait le plus petit élément de chaque ligne (ici 1) pour faire apparaître un « 0 » et idem en colonne s'il n'y a pas un « 0 » par ligne et par colonne (ici 2 pour d et e). L'ordre n'importe pas. Le coût de la satisfaction est donc ici au minimum de (5) si tout va bien à la première étape puis de (9) à la fin de la deuxième étape et (11) par l'affectation de D. Le tableau devient :

	a	b	c	d	e
A	0	1	2	3	4
B	0	3	1	4	2
C	2	1	0	4	3
D	0	1	2	4	3
E	1	0	3	2	4

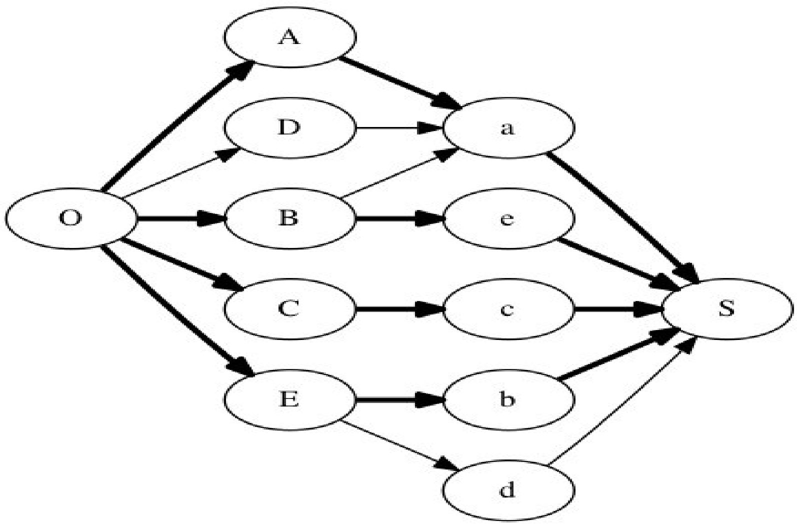
Puis

	a	b	c	d	e
A	0	1	2	1	2
B	0	3	1	2	0
C	2	1	0	2	1
D	0	1	2	2	1
E	1	0	3	0	2

Le début des affectations en commençant par (A) donne le tableau suivant :

	a	b	c	d	e
A	0				
B					0
C			0		
D					
E		0			

Ceci donne une première affectation (équivalent au marquage initial d'un réseau dans Ford-Fulkerson). Sur un réseau les arcs sont tous de capacité 1, les noeuds du graphe bi-parti sont les élèves et les chambres. Le réseau devient donc (arc en gras saturé, en fin non saturé):



Il n'existe pas de chaîne, donc le réseau est avec un marquage maximal, il n'est pas possible d'affecter un 0 supplémentaire.

(A,a) (B,e) (C,c) (D,d) (E,b) -> 1 + 3 + 1 + 5 + 1 = 11

Passage à la méthode hongroise (à partir du tableau d'affectations initiales):
Marquer les lignes n'ayant pas de 0 encadré (ici D en gras).
Marquer toutes les colonnes ayant un « 0 » barré sur une ligne marquée (ici a en gras).

> (D) : lignes n'ayant pas de 0 encadré : lignes non affectées
>
> (a) : colonnes n'ayant pas un ∅ sur une ligne marquée : alternative d'affectation initiale.

	a	b	c	d	e

A	0				
B	∅				0
C			0		
D	∅				
E		0		∅	

Marquer toutes les lignes ayant un « 0 » encadré dans une colonne marquée (ici A en gras)
Et revenir à l'étape (b) jusqu'à ce que le marquage ne soit plus possible (c'est le cas).
(A) affectation à remettre en cause

	a	b	c	d	e
A	0				
B	∅				0
C			0		
D	∅				
E		0		∅	

Barrer les lignes non marquées et les colonnes marquées (ici B, C, E et a)
Cela revient à éliminer les non impliqués par les remises en cause

	~~a~~	b	c	d	e
A	**~~0~~**	**1**	**2**	**1**	**2**
~~B~~	~~∅~~	~~3~~	~~1~~	~~2~~	~~0~~
~~C~~	~~2~~	~~1~~	~~0~~	~~2~~	~~1~~
D	~~∅~~	1	2	2	1
~~E~~	~~1~~	~~0~~	~~3~~	~~∅~~	~~2~~

Retirer le plus petit nombre restant (ici 1) de tous les éléments non barrés et ajouter le aux éléments barrés deux fois (ici Ba, Ca, Ea).
Définition du coût minimum à payer. Vient en surplus à tous les éléments associés à la remise en cause mais non retenus.

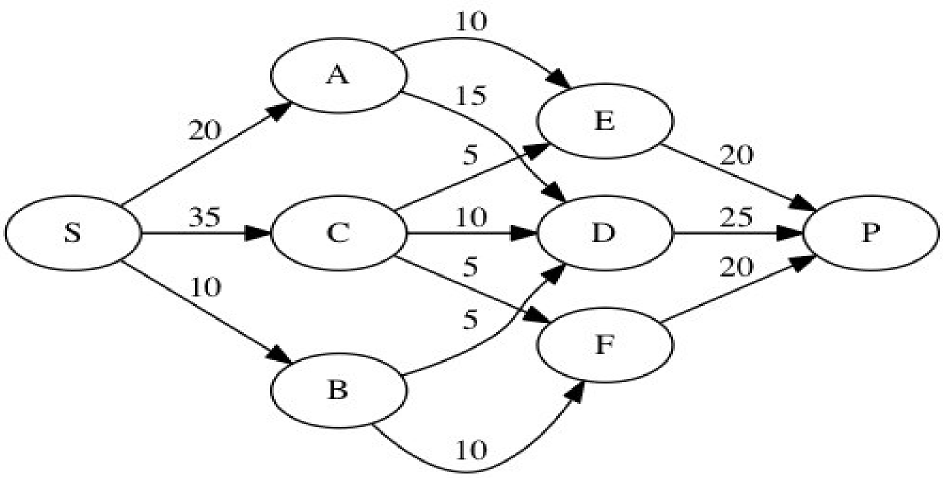
	a	b	c	d	e
A	0	0	1	0	1
B	1	3	1	2	0
C	3	1	0	2	1
D	0	0	1	1	0
E	2	0	3	0	2

A partir de ce tableau on peut affecter un 0 par ligne et colonne. Le problème est donc réglé.
Les trois solutions possibles sont donc :

(A, a), (B, e), (C, c), (D, b), (E, d) = 1 + 3 + 1 + 2 + 3 = 10 ou
(A, b), (B, e), (C, c), (D, a), (E, d) = 2 + 3 + 1 + 1 + 3 = 10 ou
(A, d), (B, e), (C, c), (D, a), (E, b) = 4 + 3 + 1 + 1 +1 = 10

7.4 Dépôts de marchandises

Le problème considéré ici est une gestion de flux. Le graphe associé est :



Résolution : On constate que F est approvisionné directement par BF : 10 et CF : 5 pour un total de 15. Il y aura donc pénurie et B sera vide. Dans une deuxième étape il est possible de saturé D par AD : 15 et CD : 10. Il reste donc à satisfaire E sur la base de AE : 5 et CE : 5. A devient vide et C a saturé ses capacités de sorties. Il ne peut y avoir d'amélioration (famine).

7.5 Tout bénéfice

L'objectif sera donc d'optimiser le revenu généré par les différents produits. Soient p1, p2 et p3 les nombres d'unités produites de P1, P2 et P3 respectivement. Les contraintes sont : une production inférieure ou égale à la capacité d'absorption du marché pour chacun des produits et un temps de production compatible avec la base de 45h.

$$\begin{aligned}
&\text{Max } 4\,p1 + 12\,p2 + 3\,p3 \\
&\quad 0 \le p1 \le 1000 \\
&\quad 0 \le p2 \le 500 \\
&\quad 0 \le p3 \le 1500 \\
&\quad 1/50\,p1 + 1/25\,p2 + 1/75\,p3 \le 45
\end{aligned}$$

7.6 Programmation dynamique

Graphe : noeud = couple (début du mois, valeur du stock pendant ce mois). On évalue les différentes possibilités (programmation dynamique déterministe).

Arête : coût généré

L'objectif est de construire un graphe dont la valeur des nœuds sera le minimum des coûts de production et de stockage. La production va dépendre du stock déjà disponible pour arriver à un stock nul au mois d'avril.

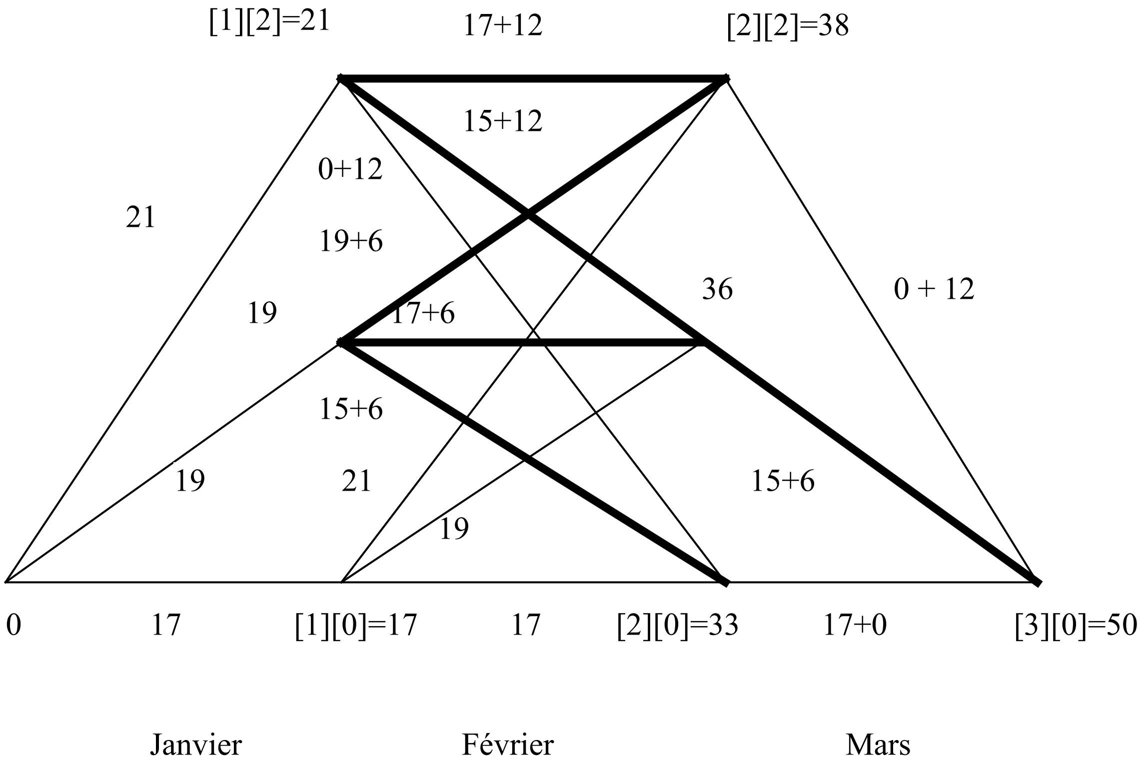
Par exemple :
A la fin du mois de janvier avec un stock nul ([1][0]) (on a produit 2 unités et utilisé ces deux unités) -> [1][0] : [0][0] + C(2,0) = 0 + f(2) + 6 * 0 = 0 + 17 + 0 = 17
A la fin du mois de janvier avec un stock de 1 ([1][1]) (on a produit 3 unités et utilisé 2 unités mais pas de coût de stock) -> [1][1] : [0][0] + C(2,0) = 0 + f(3) + 6 * 0 = 19
A la fin du mois de février avec un stock de 2 unités (on en a produit 4 et consommé 2) :
Venant de [1][0] : C(4,0) = [1][0] + production de 4 (f(4)) et 0 de stock

C(4,0) = f(4) + 6*0 = 21 + 0

Coût en [2][2] : [1][0] + 21 = 38

Venant de [1][2] : C(2,2) = Production de 2 et stock de 2 (f(2) + 6 * 2) = 17 + 12 = 29

Coût en [2][2]= [1][2] + 29 = 21 + 29 = 50 (supérieur à 38 donc non retenu)



Abscisses : Mois, Ordonnées : Stock

7.7 Picsou Magazine

a) Graphe : les nœuds les points de contact, les arêtes (ici on est non orienté car les traits peuvent être dessinés dans un sens ou dans un autre) sont les traits. On cherche à passer par les arêtes une fois et une seule mais par toutes les arêtes. Le problème revient donc à chercher une chaîne eulérienne.
b) Ne pas lever le crayon impose donc que le graphe soit obligatoirement connexe. Le théorème d'Euler s'applique. On ne peut tracer une forme que si le nombre de noeuds de degré impair (i.e., le nombre de point de contact) est de 0 ou 2.

7.8 Train-Train habituel

Le problème formel est la représentation d'un graphe planaire puisque les arêtes ne doivent pas se couper. Le graphe est biparti (deux ensembles de nœuds X et Y) donc il ne peut pas y avoir de liaison entre des nœuds appartenant à X (resp. Y). Pour faire une face, il faut donc au minimum 4 arêtes (X.1 vers Y.1, Y.1 vers X.2, X.2 vers Y.2 et Y.2 vers X.1 pour fermer la face). Toute face ayant un nombre d'arêtes supérieur à 4 aura obligatoirement un nombre d'arêtes paire.

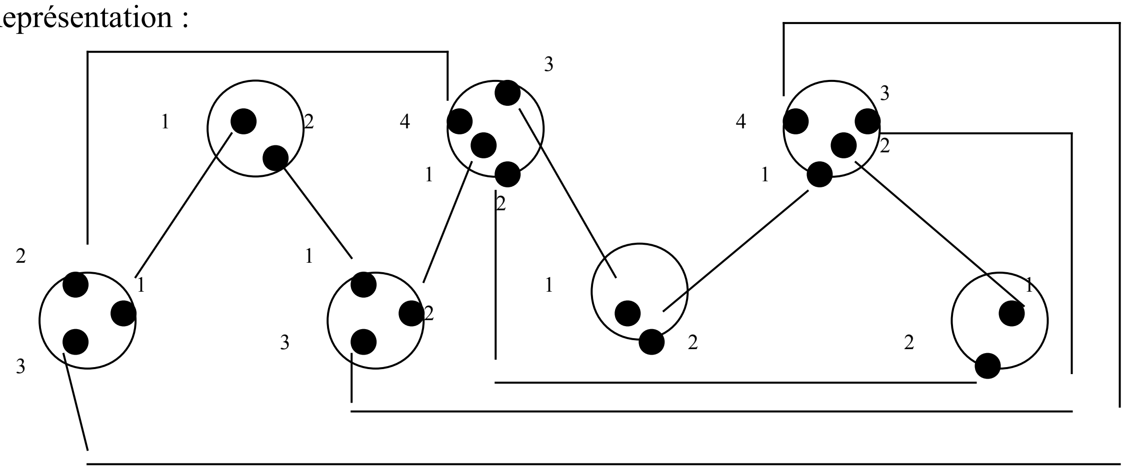
Le nombre de face est supérieur ou égal à 2. Il y a donc au plus une face infinie et au moins une face finie.

On recherche une relation entre f et m. Soit mi le nombre d'arêtes entourant la face i donc mi ≥ 4 et $\sum$ (i= 1… f) mi ≥ 4 f

Dans le cadre d'une face finie, une arête sépare uniquement deux faces donc chaque arête est comptée au plus deux fois dans la détermination de la somme. Pour les arêtes pendantes elles ne sont comptées qu'une fois dans la face infinie ou dans la face finie.

$\sum$ (i= 1… f) mi $\leq$ 2m   Donc 4 f $\leq \sum$ (i=1 ..f) mi $\leq$ 2 m => f $\leq$ m/2

Représentation :



7.9 Et glou et glou et glou

Graphe :

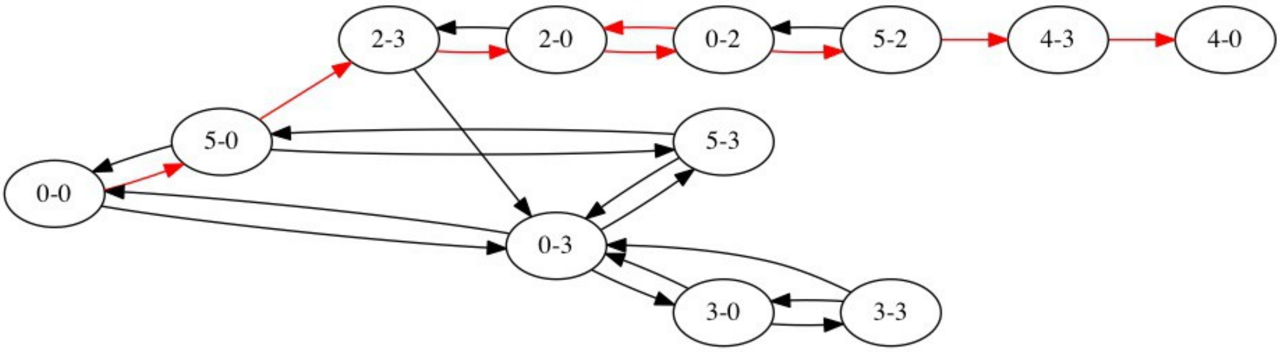
Les noeuds sont des couples donnant le contenu du re cipient de 5 litres et celui du re cipient de 3 litres.

Les arcs (car orienté) représente le passage par transvasement soit d'un récipient vers un autre soit à partir de la cuve initiale.

On place un arc entre deux noeuds lorsqu'on peut passer d'une configuration a l'autre.

Problème : On cherche alors un chemin du noeud (0,0) au noeud (4,0).

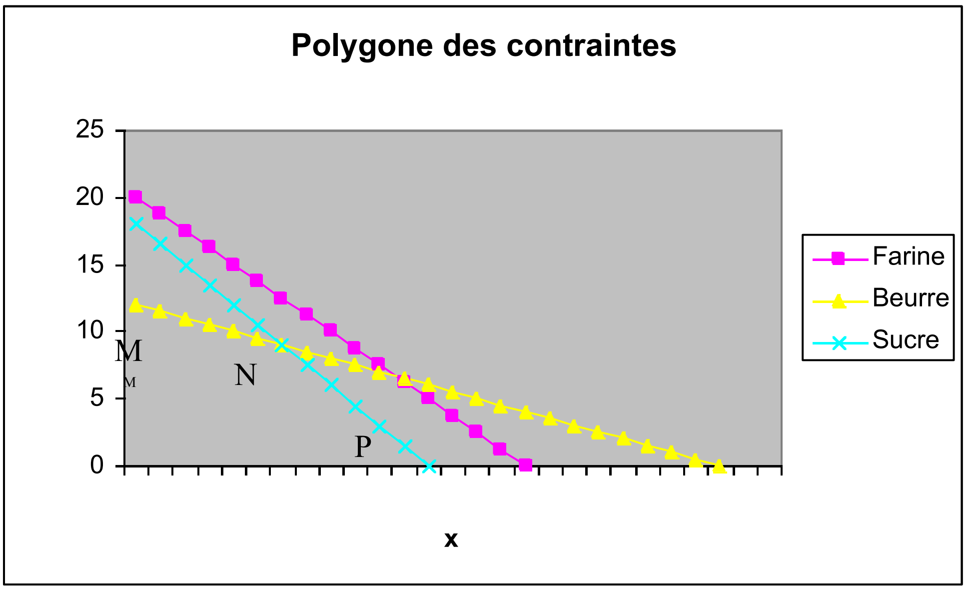
Sous-ensemble du graphe :



7.10 Boulanger

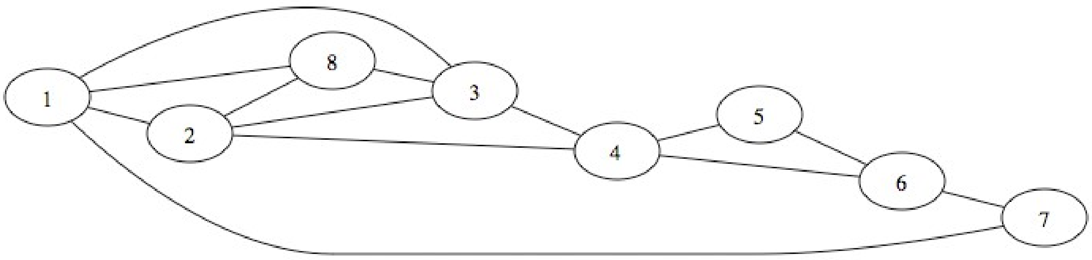
Le polygone des réalisables est limité à l'intérieur de OMNP. Le problème devient donc sur le pourtour de ce polygone quelles sont les valeurs de x et de y qui optimisent la fonction des ventes. La maximisation des ventes s'effectue sous la forme d'une fonction R : CA = 40 x + 50 y où x représente le nombre de brioche et y le nombre de pain. Sur le graphique la droite R se déplace sur un optimum de x = 6, y = 9 et CA = 690

Les inéquations sont :	Pour la farine :	$5x + 4y \leq 80$
	Pour le beurre :	$1x + 2y \leq 24$
	Pour le sucre :	$3x + 2y \leq 36$



7.11 Complet

1) Graphe d'ordre 8 : les noeuds sont les pays et les arêtes les frontières.



2) Complet : non manque (5,1) par exemple; Connexe : oui, il n'existe pas de noeud qui ne puisse pas être joignable à partir de n'importe quel autre noeud.
Degrés : d(1) = 4; d(2) = 4; d(3) = 4; d(4) = 4; d(5) = 2; d(6)=3; d(7) = 2; d(8) = 3;
Nombre d'arêtes : Somme des degrés / 2 = 13

3) Distance = la longueur de la chaîne la plus courte. Pas d'arc direct entre 1 et 5, 5 n'est pas un noeud d'un arc adjacent à un successeur de 1, donc le minimum de la longueur est (1, 7), (7, 6), (6, 5) par exemple donc distance = 3.
Diamètre = la longueur la plus longue de chemins entre deux noeuds quelconques différents du graphe.

	1	2	3	4	5	6	7
1	X	1	1	2	**3**	2	1
2		X	1	1	2	2	2
3			X	1	2	2	2
4				X	1	1	2
5					X	1	2
6						X	1
7							X

4) Partir d'un chemin après avoir franchi chaque frontière = une chaine eulérienne, et revenir à son point de départ = cycle eulérien
Chaîne eulérienne : le nombre de noeud de degré impair est égal à 0 ou 2
ici : noeud 6, noeud 8 → réponse Oui
Cycle eulérien : tous les noeuds sont de degré pair
ici : contre-exemple 6 → réponse Non

5) nombre maximum de pays sans frontière commune : On prend ici le graphe complémentaire (deux noeuds sont reliés s'ils n'ont pas de frontière commune). Tout pays ayant une frontière commune avec tous les autres pays ne figurera donc pas dans ce graphe. On cherche un stable de cardinalité maximum.

7.12 Embouteillage

Problème de flot maximal (Ford-Fulkerson) :

U2 – PF3 (20/22) – PF5 (20/22) – C2 (**10/10**) - C3 (**10/10**)
U2 – PF2 (5/6) – C2 (5/15) : C2 soldé
U1 – PF2 (10/15) – C3 (10/15) : C3 soldé
U1 – PF1 (10/20) – PF2 (**10/10**) – C1 (**10/10**)
U1 – PF1 (5/10) – PF4 (5/15) – C1 (5/7) : C1 soldé

7.13 Vente d'ordinateurs

Le noeud s correspond à la centrale d'achat, le noeud p correspond aux clients. Pour chaque période on introduit deux noeuds i' et i". Pour chaque période on introduit un arc (s, i') avec une
capacité infinie et un coût pi (cela correspond à l'achat). A chaque période on introduit un arc (i', i") avec une capacité de B (pas plus de B PC en stock) et un coût hi (arc de stockage). A chaque période on introduit un arc (i", p) avec une capacité yi (on ne peut pas vendre plus de yi PC, c'est l'arc de vente) et un coût nul. A chaque période (sauf la dernière), on introduit un arc i" vers le i' de la période suivante avec une capacité infinie et un coût nul. L'objectif est de chercher un flot compatible de coût minimum.

7.14 Chèques en bois

1) A quel problème cela correspond-il ? → un arbre de poids minimum
2) Donnez la modélisation
   Graphe non orienté, avec noeud = un bosquet, une arête = une route,
   on doit pouvoir aller à tous les bosquets = graphe connexe,
   Pour minimiser le coût de construction = pas de cycles donc connexe, sans cycle = arbre
3) Donnez la solution :
   Trier les arbres par ordre de poids croissant et poids = 10,4
      avec (7,8), (6,7), (1,5), (4,5), (5,8), (2,3), (3,8)

7.15 In vino veritas

Modélisation
   2 catégories : Homme (H) et Femme (F)
   2 vêtements : Pantalons (P), Short (S)
   Nombre de personnes (NB) = FS + FP + HS + HP
Combien y a-t-il de personnes ?
   NB = 100 personnes
      FS + FP = 8
      FP = 4 => FS = 4
      FP + HP = 19 => HP = 15
      HS = 77
      NB = 4 + 4 + 77 + 15

7.16 Les adieux

Donnez la modélisation : les noeuds sont les personnes, les arêtes (relation symétrique) sont les serrements de mains, pas de boucle (on ne se serre pas la main), chaque personne est reliée à (n - 2) autres personnes. Une personne ne serre pas la main de la personne avec laquelle elle part.
Solution : $n * (n - 2) / 2 = 112 \Rightarrow n = 16$

7.17 Les carrés

Ce graphe est en fait constitué de deux sous-graphes isomorphes reliés entre eux. Chaque sous-graphe à des cycles de longueur paire. Pour passer d'un sous-graphe à l'autre et revenir au premier sous-graphe il faut obligatoirement deux arêtes. Tout cycle est donc obligatoirement de longueur paire.

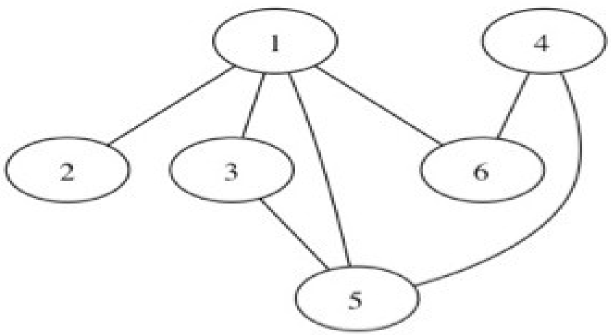
C'est un graphe bi-parti :(1, 4, 5, 8) (7, 2, 3, 6)

7.18 Les triages

L'ordre d'entrée en gare est défini, la relation est donc symétrique donc non orienté

Noeud : wagon. Arête : relie deux noeuds *i* et *j* si les wagons *i* et *j* ne peuvent pas être sur la même voie de triage.

Le graphe est le suivant :



Le problème : Détermination du nombre chromatique. (Chacun des wagons s'excluant mutuellement d'être sur la même voie, cela correspond au nombre de voies nécessaire).

4 3 2	6 5 2	5 2	6 3 2
6 5	4 3	6 3	5
1	1	4 1	4 1

Attention d'une manière générale : le nombre chromatique est différent de l'ordre du sous-graphe maximal (exemple un cycle de longueur 5 : le nombre chromatique est 3, l'ordre du sous-graphe complet maximal est 2).

Un ordre d'arrivée du type : 5, 2, 1, 3, 4 conduit à un cycle de longueur 5 (a,b,e,d,c) avec a=1, b=2, e=5, d=4 , c=3) et donc trois voies (e), (cb), (ad).

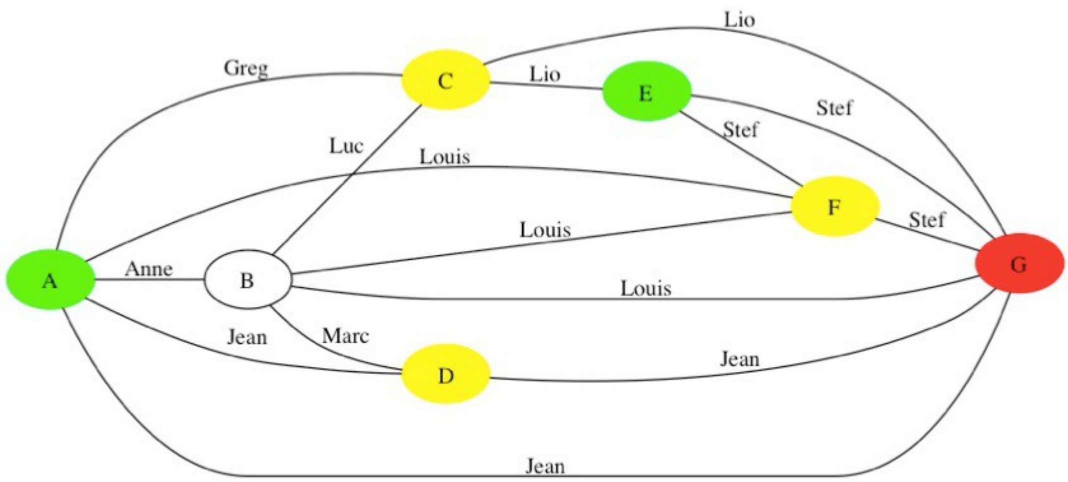
Idem pour une arrivée dans l'ordre : 1, 2, 3 → 1 voie, nombre chromatique = 1, ordre = 0

7.19 Festival à Cannes

Modélisation des incompatibilités de tournage → non orienté, Noeud : film, Arête : incompatibilité car partage au moins une personne

Problème : déterminer le nombre chromatique du graphe (chaque film de même couleur pourra être tourné en même temps). On cherche donc à minimiser le nombre de couleur.

Graphe



Nombre : 4 à cause du sous-graphe complet {A, B, C, G}

E peut être en vert ou en bleu.

7.20 Flou hamiltonien

Définition : voir cours

Réponse oui. A partir d'un algorithme de type glouton, on prend un nœud non déjà retenu au hasard à partir d'un ensemble initial de nœuds retenus vide. Le graphe est complet, il est donc obligatoirement relié à un autre nœud non encore retenu à partir du moment ou l'ensemble des nœuds non retenus n'est pas vide. Ce nœud est donc le suivant visité. L'application récursive de ce processus conduit à définir une chaîne passant par tous les nœuds.

7.21 Attention travaux

Les noeuds sont les carrefours. Les rues sont à double sens donc graphe non orienté (arêtes). Les arêtes sont les rues. Les rues ne débouchent pas sur le même carrefour donc graphe simple. Le graphe est simple et connexe (par hypothèse). Tous les nœuds sont de degré pair (par hypothèse). Il existe donc un cycle eulérien. La suppression d'une arête de ce cycle transforme le cycle eulérien en chaine eulérienne. Une chaine eulérienne est un graphe connexe.

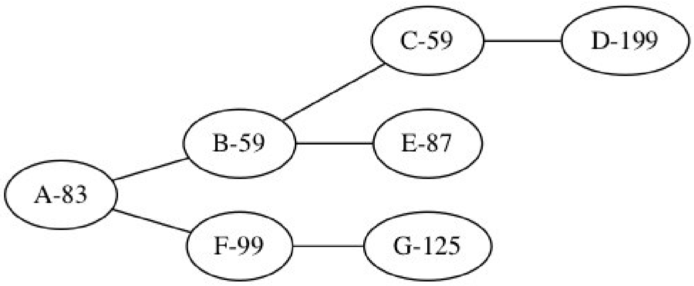
La connexité n'est donc pas perdue.

7.22 Attention danger enfant

Principe :
Pour chaque nœud i : On va calculer pour chaque nœud, j, de N (# de i) le coût du chemin associé de j à i. On le multiplie par le nombre d'enfant qui doivent effectuer ce trajet (K(i)). On somme tous ces coûts de chemins. Au final on prend le nœud pour lequel cette somme est minimale.

$$\underset{i \in N}{\mathrm{Min}} \left(\underset{j \in N}{\overset{i \# j}{\Delta}} \sqrt{} \, (\mathrm{Path}\,(j, i, G), \mathrm{coutChemin}) * K(j) \right)$$

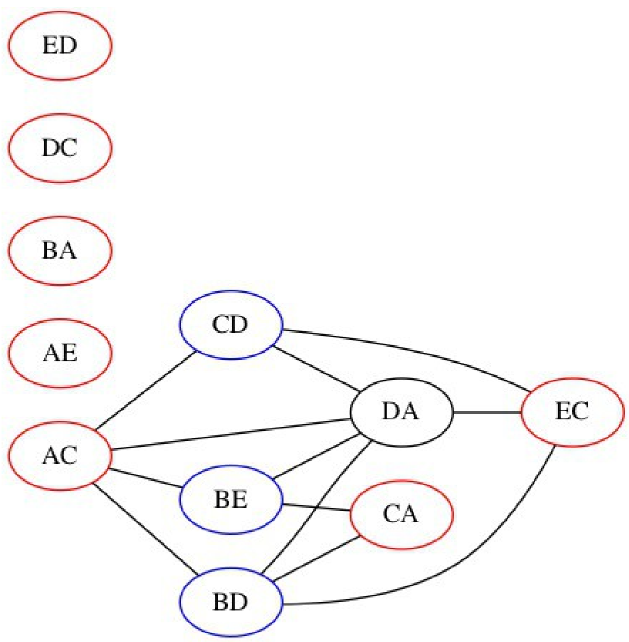
83 = 2 x 8 + 2 x 3 + 7 x 2 + 4 x 9 + 1 x 5 + 2 x 3



En l'occurrence ici soit B soit C.

7.23 Le carrefour des possibles

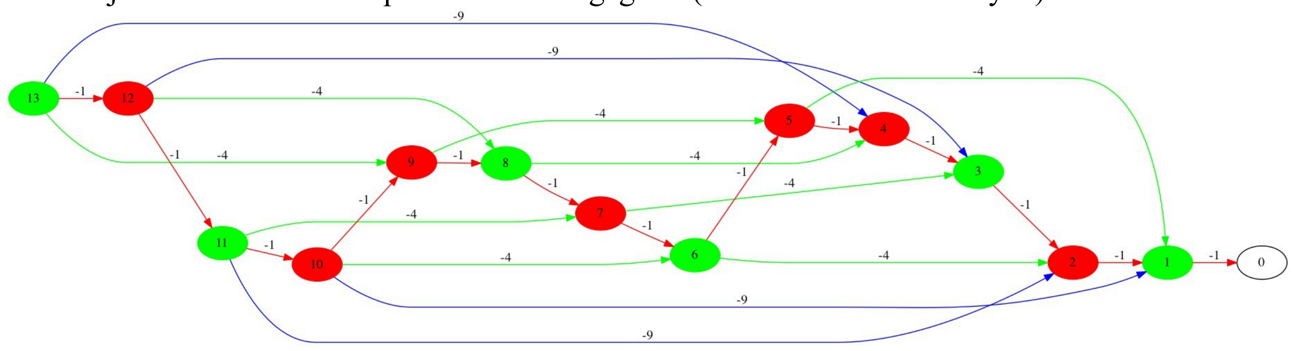
Les nœuds représentent les trajets possibles sous la forme de couple de rues. Les arêtes modélisent les incompatibilités. Le nombre de couleur nécessaires pour colorier le graphe définit le nombre de cycle de feux tricolores.
K3 => 3 couleurs minimum et en fait suffisantes.



7.24 Sac de billes

1) Justifiez que l’un des deux joueurs possède forcément une stratégie gagnante, quel que soit n

Graphe : Noeud : nombre de bille, Arcs : le passage d'un nombre de bille à un autre conformément au retrait d'un nombre de billet qui est un carré.
Il n'y a pas de remise le graphe est donc acyclique. Chaque joueur est obligé de jouer donc il ne peut pas y avoir de boucle. Graphe acyclique sans boucle => noyau unique. Il existe une stratégie qui permet à un joueur de gagner. Pour construire le noyau : un tri topologique.
.
2) Construire le graphe et montrer quel joueur doit commencer pour être sur de gagner si n = 13.
Si le joueur ne commence pas il est sur de gagner. (les verts forment le noyau).



7.25 Pic et Pic et Colégram

(a) Modélisation
Noeuds : étudiants
Non orienté : si a est à coté de b alors b est à coté de a

Arête : est à coté de
Graphe : ici un K_9

(b) Une réunion correspond à un cycle hamiltonien de K_9 (un cycle passant une et une seule fois par chaque noeud).
Si deux cycles correspondant à deux réunions ont une arête commune, cela signifie que les deux étudiants reliés par cette arête sont voisins. Ainsi, le problème revient à déterminer le nombre de cycles hamiltoniens disjoints de K_9.

(c) Un K_9 a n * (n-1) / 2 arêtes => 9 x 8 / 2 = 36 arêtes.
Chaque cycle utilisant 9 arêtes, ce nombre est au maximum égal à 4.
Exemple : 1,2,3,9,4,8,5,7,6 — 1,3,4,2,5,9,6,8,7 — 1,4,5,3,6,1,7,9,8 — 1,5,6,4,7,2,8,1,9