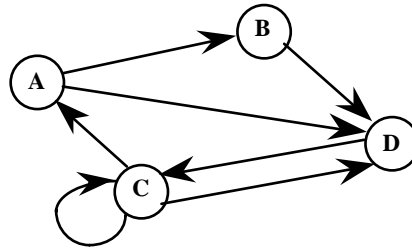


# **Travaux Dirigés**



Exercice 1 :

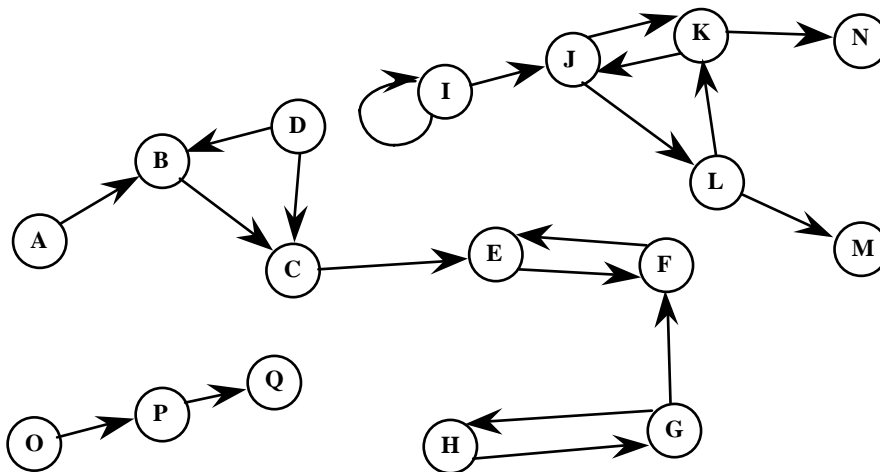
Soit le graphe



- 1) Enumérer:  $U(A)$ ,  $U(B)$ ,  $U^-(C)$ ,  $U^+(D)$ .
- 2) Calculer:  $d^+(C)$ ,  $d^-(C)$ ,  $d(C)$ .
- 3) Rapporter un chemin simple mais pas élémentaire.
- 4) Rapporter un circuit Hamiltonien.

Exercice 2 :

On considère le graphe  $G=(X,U)$  ci-dessous. Rapporter des chaînes de B à G de longueur 7, 8, 9, 10 et 11. Déterminer ses composantes connexes, ses composantes fortement connexes ainsi que le graphe réduit.



Exercice 3 :

Montrer le lemme de KOENIG : on peut toujours extraire d'un chemin allant de  $i$  à  $j$  un chemin élémentaire allant de  $i$  à  $j$ . Généraliser à une chaîne.

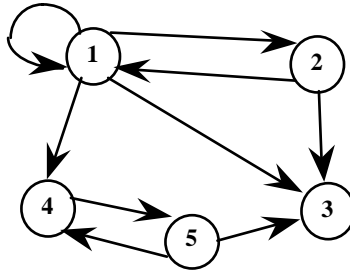
Exercice 4 :

Quelles méthodes proposez-vous pour coder un graphe en machine ?

### Exercice 5

On propose deux méthodes pour coder un graphe  $G=(X,U)$  en machine : la matrice d'adjacence  $A$  et la file des successeurs de tableaux ALPHA1, ALPHA2 et BETA. La matrice associée est définie par  $A=(a_{ij})$  avec  $a_{ij}=1$  si l'arc  $(i,j)$  appartient à  $U$  et  $a_{ij}=0$ , sinon. La file des successeurs BETA est le tableau des successeurs, les successeurs du sommet  $I$  se trouvant entre les adresses ALPHA1(I) et ALPHA2(I) dans le tableau BETA, (en cours on avait un seul tableau ALPHA).

Rapporter les tableaux  $A$ , ALPHA1, ALPHA2 et BETA pour le graphe ci-dessous.



Ecrire un algorithme permettant de passer de la file des successeurs à la matrice associée. Quelle est la complexité de cet algorithme?

Ecrire un algorithme permettant de passer de la matrice associée à la file des successeurs. Quelle est la complexité de cet algorithme ?

remarque:

On a en fait deux matrices associées car les valeurs peuvent être entières ou booléennes.

### Exercice 6 :

Donner le nombre d'arêtes d'un graphe non orienté complet de  $n$  sommets.

### Exercice 7 :

Expliquer pourquoi si  $G=(X,E)$  est un graphe non orienté sans boucle, la somme des degrés est égale à deux fois le nombre d'arêtes du graphe.

### Correction exercice 5 :

```
{ file des successeurs vers matrice d'adjacence }
for i := 1 to N do
for j := 1 to N do
A[i,j] := 0;
for i := 1 to N do
begin
j1 := ALPHA1[i];
j2 := ALPHA2[i];
if ( j1 <> 0 ) then
begin
for j := j1 to j2 do
A[i, BETA[j]] := 1;
end;
end;
end;
```

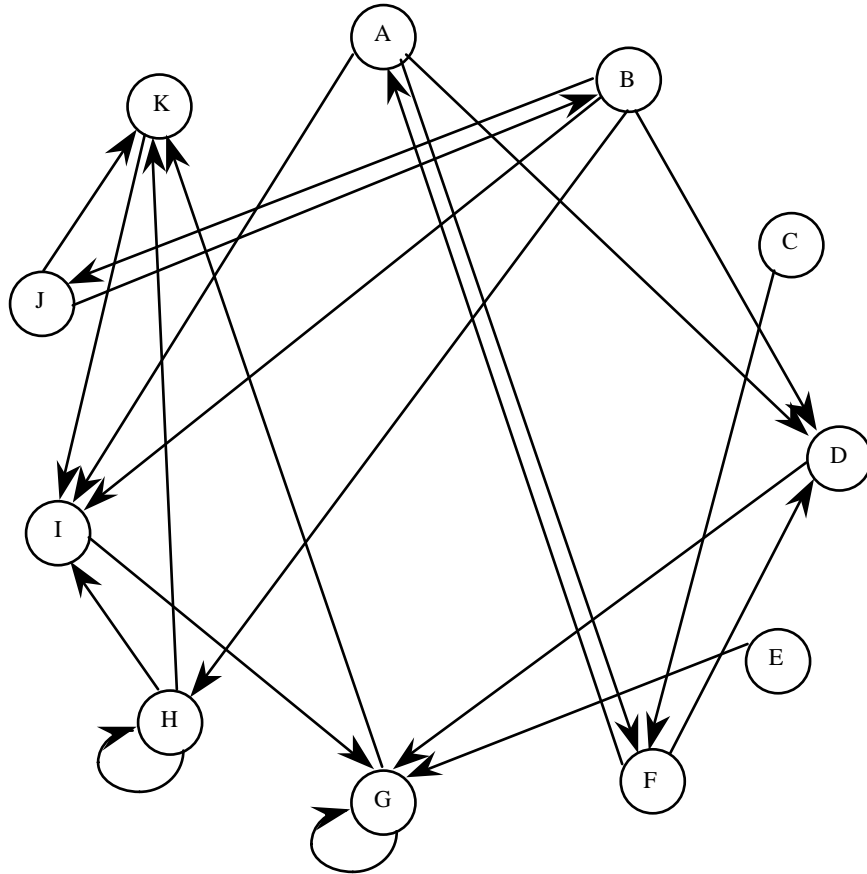
```
{ matrice d'adjacence vers file des successeurs }
```

```
indi := 1;
for i :=1 to N do
begin
j1 := 0;
j2 := 0;
for j := 1to N do
if A[i,j] = 1 then
begin
if j1 = 0 then j1 := indi;
BETA[indi] := j;
j2 := indi;
indi := indi + 1;
end;
ALPHA1[i] := j1;
ALPHA2[i] := j2;
end;
end;
```



**Fermeture transitive d'un graphe  $G=(X,U)$  orienté et composantes fortement connexes.**

On considère le graphe suivant :



Définition 1 : Soit  $i$  un sommet de  $G$ . On appelle fermeture transitive directe de  $i$ , l'ensemble des sommets  $j$  de  $G$  tels que :  $(i=j)$  ou (il existe un chemin de  $i$  à  $j$ ),  $j$  est alors un descendant de  $i$ .

Question 0 : rapporter la matrice associée à ce graphe.

Question 1: détermination des successeurs d'un sommet

On note  $\Gamma(x_i)$  l'ensemble des successeurs du sommet  $x_i$ . Ecrire les successeurs des sommets du graphe ci-dessus.

Quelle est la complexité de la recherche des successeurs d'un sommet si le graphe est codé par la matrice associé ?

Question 2 : calcul des descendants d'un sommet

On note  $\widehat{\Gamma}(i)$  l'ensemble des descendants de  $i$  ( $i$  est considéré comme descendant d'ordre 0 de lui même), et  $\Gamma^k(i) = \Gamma(\Gamma^{k-1}(i))$  l'ensemble des successeurs de  $i$  de l'ordre  $k$ .

Montrer que :  $\widehat{\Gamma}(i) = \{i\} + \Gamma(i) + \Gamma^2(i) + \Gamma^3(i) + \dots + \Gamma^{n-1}(i)$ .

On suppose que le graphe est codé par la matrice associée. Ecrire les descendants du sommet A du graphe ci-dessus ?

Quelle est la complexité, en utilisant cette méthode, de la recherche des descendants d'un sommet si le graphe est codé par la matrice associée ?

### Question 3 : calcul des descendants d'un sommet

Soit j un descendant de i. On dit que j est à distance k de i si k est la longueur du plus court chemin entre i et j, on va dire que j appartient à  $D^k(i)$ .

Montrer que :  $\widehat{\Gamma}(i) = \{i\} + D^1(i) + D^2(i) + \dots + D^k(i) + \dots + D^{n-1}(i)$ .

Quelle est la complexité, en utilisant cette méthode, de la recherche des descendants d'un sommet si le graphe est codé par la matrice associée ?

Définition 2 : On appelle fermeture transitive inverse de i, l'ensemble des sommets j de G tels que  $(i=j)$  ou (il existe un chemin de j à i) j est alors un ascendant de i.

### Question 4 : Détermination des prédécesseurs d'un sommet

On note  $\Gamma^{-1}(i)$  l'ensemble des prédécesseurs du sommet  $x_i$ .

Ecrire les prédécesseurs des sommets du graphe ci-dessus.

### Question 5 : Détermination des ascendants d'un sommet

Proposer une méthode analogue à la précédente pour calculer les ascendants d'un sommet i. Quelle est sa complexité ?

### Question 6 :

Que représente l'ensemble des sommets qui sont à la fois descendants et ascendants du sommet i ?

### Question 7 :

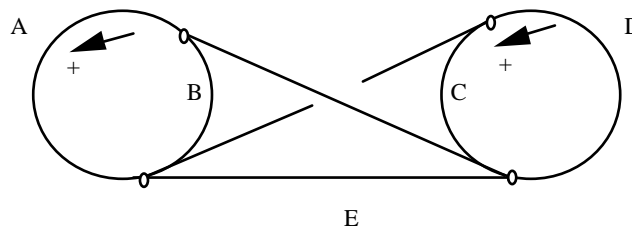
Proposer un algorithme basé sur les questions précédentes pour chercher l'ensemble des composantes fortement connexes d'un graphe. Quelle est sa complexité (cette méthode s'appelle la méthode de MALGRANGE) ? Appliquer cette méthode à l'exemple précédent en utilisant la matrice écrite. On distinguera, en appliquant l'algorithme, les descendants (resp. ascendants) d'ordre 1, d'ordre 2, ... etc.

### Question 8 :

Dessiner le graphe réduit, le graphe initial est-il planaire ?

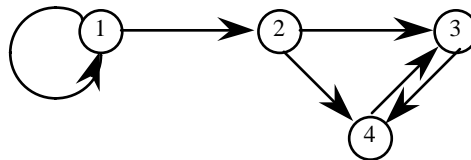


Exercice 1 : Soit un train électrique dont le circuit est le suivant :



Chaque aiguillage a 2 ou 3 positions. On a remarqué qu'au bout d'un certain temps, quel que soit l'état initial, le tronçon E n'est jamais emprunté par le train. Expliquer cela à l'aide d'un graphe à 10 sommets .

Exercice 2 : Soit  $G = (X,U)$  , un graphe orienté et B la matrice d'adjacence sommet-sommet associée à G. On prendra comme exemple le graphe suivant :



2.0 : Calculer  $B, B^2, B^3, B^4, (I+B), (I+B)^2, (I+B)^3, (I+B)^4$ , pour l'exemple ci-dessus, où la matrice B est booléenne (on utilisera les opérations booléennes).

2.1 : Si  $B^k$  est le produit matriciel Booléen de B par B, montrer que:

$$\{B^k\}_{ij} = 1 \text{ s'il existe un chemin de } k \text{ arcs entre } i \text{ et } j \text{ et } 0 \text{ sinon.}$$

2.2 : On définit de même  $B^k$  la puissance k<sup>ème</sup> matricielle Booléenne de B et l'on pose :

$$\widehat{B}_k = I + B + B^2 + \dots + B^k .$$

Montrer que :

$$\{\widehat{B}_k\}_{ij} = 1 \text{ s'il existe un chemin de } k \text{ arcs au plus entre } i \text{ et } j \text{ et } 0 \text{ sinon.}$$

Montrer que l'on peut définir  $\widehat{B} = \lim_{k \rightarrow \infty} \widehat{B}_k$  pour k tend vers  $\infty$ .

2.3 : On pose de même  $\widehat{C}_k = I + C + C^2 + \dots + C^k$  où  $C = B^T$ . Montrer que la limite  $\widehat{C}$  existe et que :

$$\{\widehat{C}\}_{ij} = 1 \text{ s'il existe un chemin entre } j \text{ et } i \text{ et } 0 \text{ sinon.}$$

2.4 : Dédire de ce qui précède une méthode pour décomposer G en composantes fortement connexes. Evaluer la complexité de cette méthode.

2.5 : Proposer une méthode pour chercher les composantes connexes du graphe.

2.6 : On se propose d'améliorer cette méthode.

Démontrer l'identité  $(I+B)^k = I + B + B^2 + \dots + B^k$ .

Montrer que la suite  $\{B(1) = I+B, B(k)=(B(k-1))^2 \text{ pour } k>1\}$  converge vers  $\widehat{B}$  en un nombre fini d'étapes que l'on évaluera.

En déduire un meilleur algorithme que celui obtenu en 2.4. On en précisera l'ordre.

2.7 : Que se passe-t-il si l'on remplace le produit Booléen par le produit usuel ?

### Exercice 3 :

Un graphe dont tous les sommets sont de même degré est dit régulier. Quelles sont les graphes non orientés réguliers de degré 1 ? de degré 2 ?

Exercice 4 : Combien y a-t-il des graphes orientés à quatre sommets ?

Problème Graphe Biparti et bicoloration:

On rappelle qu'un graphe  $G=(X,U)$  est biparti si toutes ses arêtes ont une extrémité dans un sous-ensemble  $X_1$  et l'autre extrémité dans  $X-X_1$ . L'algorithme BIPAR(G) a pour objet de tester si un graphe G connexe est biparti.

Algorithme BIPAR(G) :

{-1- Initialisations}

Lire un graphe connexe  $G=(X,U)$ ; {initialement aucun sommet n'est coloré}

colorer le sommet 1 en rouge;

initialiser le booléen BIPARTI à VRAI.

{-2- Examen des sommets colorés }

Tant qu'il existe un sommet i coloré et non examiné faire

Début

Si i est rouge alors

pour chaque voisin j de i faire

Début

si j est rouge alors BIPARTI=FAUX

si j est non coloré alors colorer j en vert

Fin

sinon { i est vert alors }

pour chaque voisin j de i faire

Début

si j est vert alors BIPARTI=FAUX

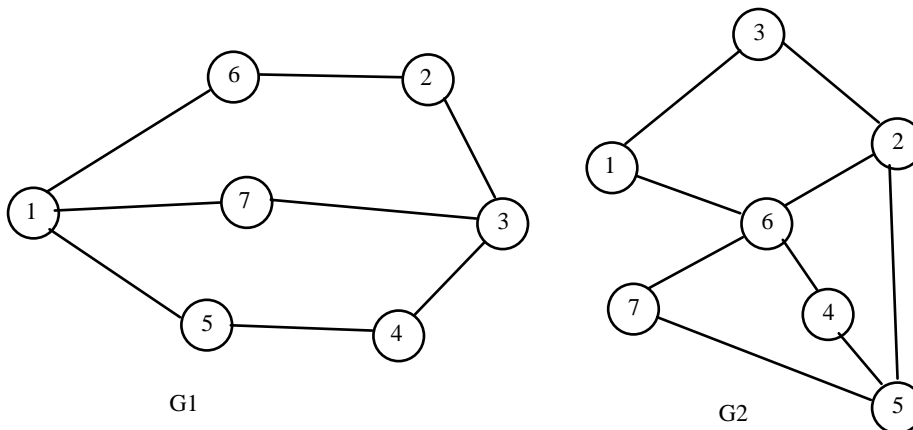
si j est non coloré alors colorer j en rouge

Fin

Fin

{-3- Sortie de l'algorithme}

Si BIPARTI = VRAI alors écrire "G est biparti", sinon écrire "G n'est pas biparti"..



Question 1: Faire tourner l'algorithme sur les graphes  $G_1$  et  $G_2$ . On rapportera les couleurs pour chaque itération de la boucle.

Question 2: On code le graphe  $G=(X,U)$  par la matrice A associée, c'est-à-dire si  $(i,j)$  est une arête du graphe alors  $a_{ij} = a_{ji} = 1$ , sinon  $a_{ij} = 0$ .

(a) Quels tableaux supplémentaires sont-ils nécessaires pour BIPAR(G) ?

(b) Quelle est la complexité résultante de l'algorithme ? Justifier.

Question 3: Proposer de façon explicite une meilleure structure de données. Que devient la complexité? Justifier.

Question 4: (Cette question est indépendante de l'algorithme)

Montrer qu'un graphe est biparti si et seulement s'il est bicolorable.

Question 5: (Cette question est indépendante de l'algorithme)

Montrer qu'un graphe biparti ne contient pas de cycle de longueur impaire.

Question 6: Quelles sont les propriétés des sommets colorés en rouge et en vert au cours de l'algorithme? Justifier. Où intervient la connexité ?

Question 7: Montrer la validité de l'algorithme. En déduire la réciproque de la question 5 dans le cas connexe.

Question 8: Comment proposez-vous de modifier l'algorithme pour tester si un graphe quelconque, c'est-à-dire non nécessairement connexe est biparti?

Question 9: Le problème de coloration minimale d'un graphe quelconque est NP-difficile. Existe-il un algorithme de complexité polynomiale permettant de le résoudre ?

Exercice 1 :

Montrez qu'un graphe est un arbre si et seulement si il existe une chaîne élémentaire et une seule entre toutes paires de sommets.

Exercice 2 :

Soit  $G$  un graphe non orienté à  $n$  sommets et  $m$  arêtes. Les arêtes de  $G$  sont notées  $u_1, u_2 \dots u_m$ .  $G_i = (V, E_i)$ , est le graphe formé des arêtes  $u_1, u_2 \dots u_i$ . On pose  $G_0 = (V, \Phi)$ .

Question 1. Quel est le nombre de composantes connexes de  $G_0$ .

Question 2. On note  $C(G_i)$  le nombre de composantes connexes de  $G_i$ . Montrer que

$C(G_i) = C(G_{i-1})$  ou  $C(G_i) = C(G_{i-1}) - 1$ .

Question 3. Interpréter le cas  $C(G_i) = C(G_{i-1})$ , puis le cas  $C(G_i) = C(G_{i-1}) - 1$ .

Question 4. Montrer qu'un graphe connexe a au moins  $n-1$  arêtes.

Question 5. Montrer qu'un graphe sans cycle a au plus  $n-1$  arêtes.

### Recherche en profondeur d'abord dans un graphe

L'algorithme ci-dessous permet de déterminer avec une faible complexité les descendants d'un sommet  $i_0$ . Initialement  $n(i)$  est le nombre de successeurs de  $i$ . On initialise également  $p(i)=0$  pour  $i$  différent de  $i_0$  et  $p(i_0)=i_0$ .

Au cours de l'exploration, on associe à chaque sommet  $i$ , un nombre  $p(i) \in [0,n]$  et on dit que **le sommet  $i$  est atteint** quand  $p(i)$  est strictement positif ( $p(i)$  est alors le sommet ayant permis d'atteindre le sommet  $i$ ) et que le sommet  $i$  est **non atteint** si  $p(i)$  est nul.

Un sommet atteint se trouvera, par convention, dans l'un des deux états suivants : **ouvert**, quand  $n(i)$  est strictement positif, et **fermé** quand  $n(i)$  est nul.

#### Algorithme :

Début

Pour  $i = 1$  à  $n$  faire {les sommets sont numérotés de 1 à  $n$ .}

    Début

$p(i) := 0$ ;

$n(i) := d^+(i)$ ;

    Fin

$p(i_0) := i_0$ ;

$i := i_0$ ;

Tant que  $(n(i_0) + \text{ABS}(i-i_0)) > 0$  faire /\*ABS(x) est la valeur absolue de x \*/

    Début

        Si  $n(i)$  différent de 0 alors {le sommet  $i$  est ouvert}

            Début

                Sélectionner parmi les successeurs de  $i$  le sommet  $j$  venant en position  $n(i)$

$n(i) := n(i)-1$ ;

                Si  $p(j)=0$  alors

                    Début

$p(j) := i$ ;

$i := j$ ;

                    Fin

            Fin

        sinon {le sommet  $i$  est fermé}

$i := p(i)$

    Fin

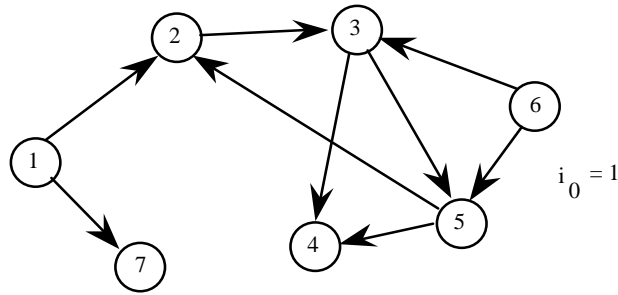
Fin

#### Première partie: étude de l'algorithme

Question 1 : On admettra provisoirement que la restriction de la fonction  $p$  à  $X-i_0-\{i/p(i)=0\}$  définit une arborescence de racine  $i_0$  à tout instant de l'algorithme.

On demande de faire tourner l'algorithme sur le graphe ci-dessous en traçant autant d'arborescences obtenues en cours d'algorithme que d'exécutions de la boucle tant que avec les conventions suivantes :

- initialement, l'arborescence est réduite à sa racine ;
- quand on sélectionne un sommet  $j$  avec  $p(j)=0$ , on ajoute l'arc "plein"  $(i,j)$  à l'arborescence;
- quand on ferme le sommet  $i$ , on met en pointillé ou en rouge l'arc  $(p(i),i)$ .



Question 2 : Expliquer intuitivement ce que l'on fait dans la boucle Tant Que. Interpréter le test :  $(n(i_0) + ABS(i-i_0)) > 0$ .

Question 3

A quoi correspondent dans la boucle Tant que les cas :

-n(i) différent de 0 ?

-n(i) égal 0 ?

Question 4

Combien de fois exécute-t-on la boucle Tant que pour l'exemple ci-dessus ?

**Deuxième partie** : preuve de l'algorithme.

Question 1: Montrer qu'en cours d'algorithme on construit une arborescence de racine  $i_0$ .

Question 2: Montrer que la restriction de cette arborescence aux arcs pleins est une arborescence, réduite à un chemin, dont  $i$  est le sommet pendant (cette arborescence est dite arborescence courante).

Question 3: A quelle condition un sommet est-il retiré de l'arborescence courante ?

Question 4: Que devient l'arborescence courante à la fin de l'algorithme ?

Question 5: Montrer que tout sommet atteint est descendant de  $i_0$ .

Question 6: Montrer que l'algorithme se termine.

Question 7: Montrer que tout sommet descendant de  $i_0$  est atteint à la fin de l'algorithme.

Question 8 : Montrer que l'algorithme détermine les descendants de  $i_0$ .

**Troisième partie** : complexité de l'algorithme

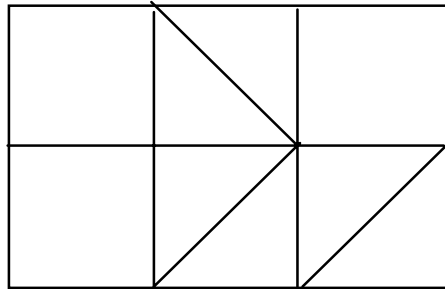
Question 1: Quel codage du graphe proposez-vous?

Question 2: Evaluer la complexité globale de l'algorithme en justifiant votre réponse.

**Quatrième partie**: extensions.

On a donc démontré que cet algorithme permet de déterminer les descendants d'un sommet  $i_0$ . Quelle méthode proposez-vous pour déterminer les composantes connexes d'un graphe? Ecrire un tel algorithme utilisant comme sous-programme l'algorithme précédent. Evaluer sa complexité.

Exercice 1 : Déterminer une chaîne eulérienne dans le graphe suivant :



Exercice 2 : Méthode arborescente :

Cinq personnes de nationalités différentes habitent les cinq premières maisons d'une rue, toutes sur le même trottoir (numérotées de 1 à 5).

Elles exercent cinq professions différentes et ont chacune une boisson et un animal favori tous différents. Les cinq habitations sont de cinq couleurs différentes.

- 1 : L'anglais habite la maison rouge.
- 2 : L'espagnol possède un chien.
- 3 : Le japonais est peintre.
- 4 : L'italien boit du thé.
- 5 : Le norvégien habite la première maison à gauche.
- 6 : Le propriétaire de la maison verte boit du café.
- 7 : Cette maison verte est immédiatement à droite de la blanche.
- 8 : Le sculpteur élève des escargots.
- 9 : Le diplomate habite la maison jaune.
- 10 : On boit du lait dans la maison du milieu.
- 11 : Le norvégien habite à coté de la maison bleue.
- 12 : Le violoniste boit des jus de fruits.
- 13 : Le renard est dans une maison voisine du médecin.
- 14 : Le cheval, à côté de celle du diplomate.

Qui élève le zèbre et qui boit de l'eau ?  
(Attribué à Lewis Carrol)

Exercice 3 :

Résoudre le problème de décodage suivant :

$$\begin{array}{r}
 \text{F O R T Y} \\
 + \quad \text{T E N} \\
 + \quad \text{T E N} \\
 \hline
 \text{S I X T Y}
 \end{array}$$





Exercice 1: On se propose de résoudre le problème de décodage suivant :

$$\begin{array}{r} \text{S E N D} \\ + \text{M O R E} \\ \hline \text{M O N E Y} \end{array}$$

Sachant que les lettres représentent des chiffres distincts compris entre 0 et 9, et que M et S ne sont pas nuls, donner la solution à l'aide d'une arborescence et montrer son unicité.

Exercice 2: Méthode de Little :( extrait de ROSEAUX)

Un représentant de commerce doit se rendre de la ville A où il réside dans quatre autres villes pour visiter les magasins clients de la marque qu'il représente. Il désire passer une et une seule demi-journée chez chacun de ses clients et revenir à sa ville de départ A. La durée du trajet entre deux villes I et J est mesurée en demi-journées et dépend de la circulation routière dans le sens IJ et JI. Le tableau donnant ces durées n'est donc pas symétrique.

$$T = (t_{ij}) =$$

	A	B	C	D	E
A	$\infty$	2	1	3	4
B	1	$\infty$	6	5	7
C	4	3	$\infty$	8	2
D	5	7	4	$\infty$	1
E	2	3	5	2	$\infty$

Sachant que ce représentant ne travaille ni ne circule le dimanche et qu'il part le premier du mois (un mercredi), quel jour au plus tôt sera-t-il de retour en A?



PROBLEME: étude du noyau d'un graphe orienté.Définition:

Un sous-ensemble  $K$  de l'ensemble  $X$  des sommets d'un graphe orienté  $G=(X,U)$  est dit noyau de  $G$  si:

1.  $K$  est un stable de  $G$ , c'est à dire : aucun couple de sommets de  $K$  n'est lié par un arc.
2. tout sommet à l'extérieur de  $K$  est extrémité initiale d'un arc dont l'extrémité terminale est dans  $K$

Discussion:

Certains graphes n'ont pas de noyau. D'autres graphes ont plusieurs noyaux. Le but principal de ce problème est de montrer qu'un graphe sans circuit a exactement un noyau et d'utiliser cette notion pour étudier des jeux.

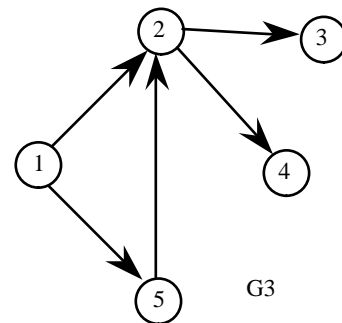
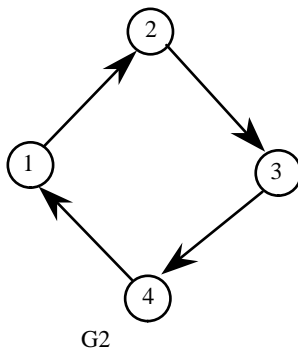
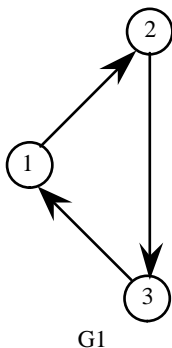
Question 0 : propriété caractéristique

Montrer que  $K$  est un noyau si et seulement si :

- tout sommet dans  $K$  a ses successeurs en dehors de  $K$ ,
- tout sommet en dehors de  $K$  a au moins un de ses successeurs dans  $K$ .

Question 1: pour se familiariser avec la notion de noyau.

Déterminer tous les noyaux (s'il en existe) des graphes  $G_1$ ,  $G_2$  et  $G_3$ .

Question 2: étude théorique des graphes sans circuit.

On se propose de montrer qu'un graphe sans circuit a exactement un noyau. Pour cela, on étudie l'algorithme NOYAU.

NOYAU

DEBUT

Lire  $G=(X,U)$  un graphe orienté sans circuit;

Initialement aucun sommet n'est coloré;

Numéroter les sommets de  $G$  selon une numérotation inverse: si  $(i,j) \in U$ , alors  $i$  est strictement plus grand que  $j$  (en particulier, le sommet 1 est un sommet sans successeur);

Colorer en rouge le sommet 1;

Pour  $l:=2,N$  faire

DEBUT

Si un des successeurs de  $l$  est rouge, colorer  $l$  en vert

Sinon colorer I en rouge

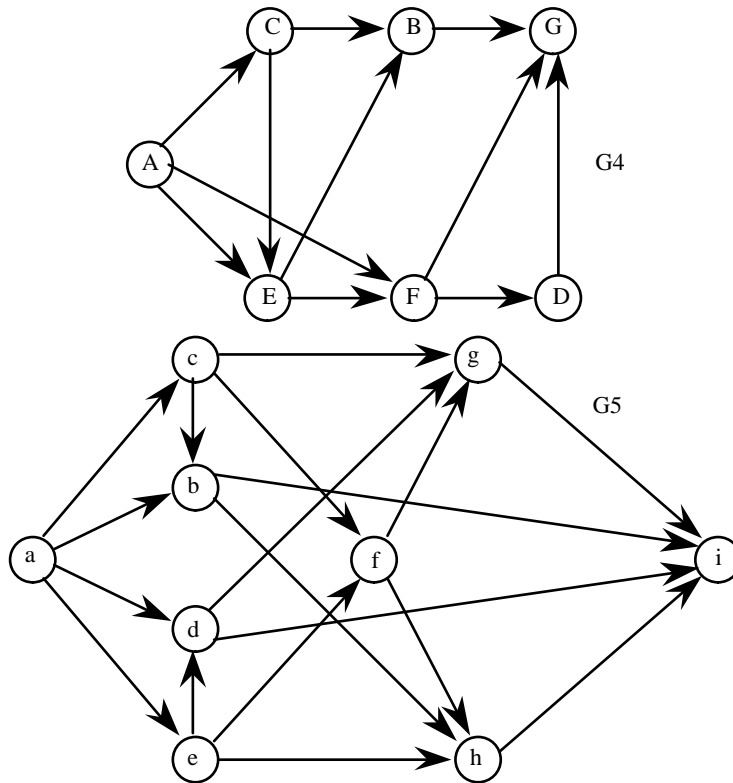
FIN

Le noyau est l'ensemble des sommets rouges

FIN

A) Faire tourner l'algorithme sur les graphes  $G_4$  et  $G_5$  ci-dessous.

On rapportera les couleurs finales.



B) Montrer qu'en cours d'algorithme :

-tout sommet vert a au moins un successeur rouge ;

-tout sommet rouge n'a pas de successeur rouge.

En déduire que les sommets rouges forment un noyau de  $G$ .

C) Montrer que l'algorithme se termine et qu'alors tous les sommets sont colorés.

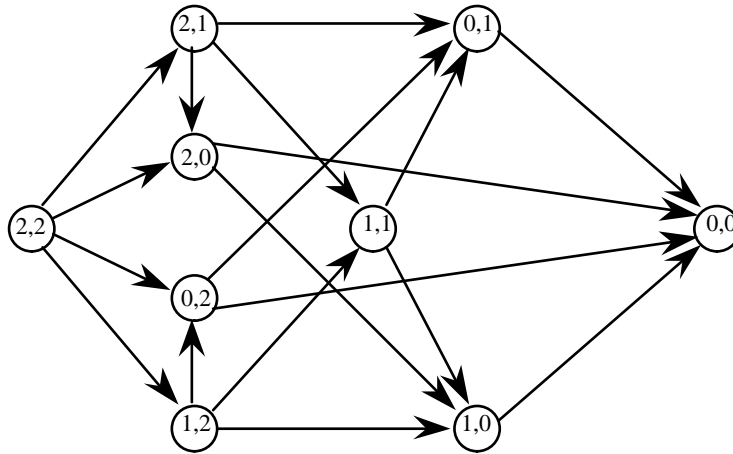
D) Montrer l'unicité du noyau.

E) Donner les tableaux principaux qui permettent de coder le plus efficacement cette procédure. On justifiera succinctement et on rapportera la complexité résultante de l'algorithme.

### Question 3: application aux jeux

Dans le jeu de NIM, on a deux piles de jetons, chacun des joueurs joue chacun son tour et prend n'importe quel nombre de jetons de l'une des deux piles. **Le joueur qui prend le dernier bâton gagne, l'autre perd.** Le jeu complet est décrit par un graphe orienté dont les sommets correspondent à l'état du jeu et les arcs aux "coups" des joueurs. Ici chaque état du jeu est décrit par une paire d'entiers  $(x,y)$  qui indique le nombre de jetons dans la première pile et la deuxième pile, respectivement.

Par exemple, si l'état initial de la pile est  $(2,2)$ , le graphe du jeu est le suivant:



La question importante pour ce jeu est: quand un joueur est-il certain de gagner et comment doit-il jouer?

Nous allons d'abord tenter de répondre à cette question pour des piles de taille initiale 2, puis on généralisera.

Une position sera dite gagnante si le joueur venant d'atteindre cette position gagne quelle que soit la façon de jouer de son adversaire sous réserve que lui-même joue bien dans la suite du jeu (c'est donc à son adversaire de jouer).

Une position sera dite perdante si le joueur ayant atteint cette position va perdre si son adversaire joue bien.

A) Quelles sont les positions gagnantes pour le cas particulier? Quelles sont les positions perdantes? Si B commence, qui va gagner?

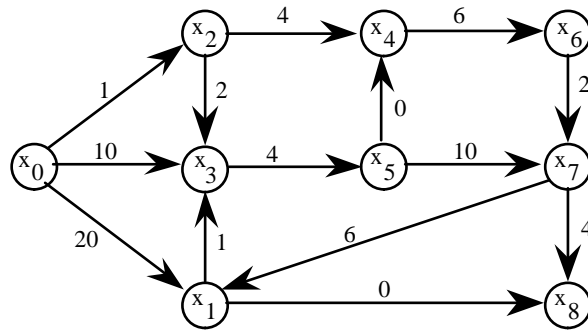
B) Pour un jeu de NIM en position de départ  $(k_1, k_2)$  et si B commence à jouer, à quelle condition sur  $k_1$  et  $k_2$ , A va-t-il gagner ? Formuler en français une stratégie gagnante. Que pouvez-vous dire si on généralise à trois piles ?

C) Le jeu de NIM est un jeu à deux joueurs, sans hasard, avec information parfaite (chacun des joueurs connaît avant de jouer l'état du jeu), à espace d'états fini et dont le graphe d'états associé est sans circuit, fini, et a un seul état final gagnant pour celui qui l'atteint, et si les deux joueurs jouent parfaitement, on peut prédire à l'avance qui gagnera à l'aide de la notion de noyau. Justifier complètement.

D) Où se situe en général la difficulté pour un tel jeu ?

E) Pourquoi la méthode précédente ne s'applique pas même théoriquement au jeu d'échecs ou au jeu de dames ? Justifier votre réponse. Quelle idée cela vous donne pour ces jeux ?



Exercice

On cherche les valeurs des chemins minimaux issus de  $x_0$ .

Les algorithmes de DIJKSTRA et BELLMAN sont-ils applicables? Justifier.

Appliquer le ou les algorithmes utilisables.

On rapportera éventuellement les valeurs successives des  $\lambda_i$  (si DIJKSTRA est utilisable) et la numérotation si elle existe, ainsi qu'une arborescence de chemins minimaux issus de  $x_0$ . Y-a-t-il unicité des chemins minimaux? Justifier.

PROBLEME : ALGORITHME DE DIJKSTRA MODIFIE

On considère un graphe valué  $G=(X,U,v)$  où la valuation  $v$  est entière de signe quelconque et où 1 est racine de  $G$ . Nous allons étudier l'algorithme suivant:

Algorithme de DIJKSTRA modifié

Début

A) Application de l'algorithme de Dijkstra:

Debut

Poser  $S:=\{1\}$ ;  $\lambda_1 :=0$ ;  $\lambda_i:=v(1,i)$  si  $(1,i) \in U$  et  $\infty$  sinon;

Tant que  $S \neq X$  faire

Début

Choisir  $i \in X-S$  tel que  $\lambda_i$  soit minimal;

Poser  $S:=S \cup \{i\}$ ;

Pour  $j \in (X-S) \cap U^+(i)$  faire  $\lambda_j:=\text{Min}(\lambda_j, \lambda_i+v(i,j))$ ;

Fin; *{Attention : lisez bien cette partie de l'algorithme,  $j \in X-S$ }*

Fin;

B) Détermination de l'arborescence de racine 1 associée :

On rappelle que cette arborescence est obtenue en retenant, pour chaque  $j$ , l'arc  $(i,j)$  ayant permis de fixer  $\lambda_j$ . Les arcs de cette arborescence  $A$  seront dits marqués.

C) Test des  $d_j$  :

Début

Chercher un arc  $(i,j)$  non marqué tel que  $\lambda_j - \lambda_i > v(i,j)$ ;

Si un tel arc n'existe pas alors FIN1;

Si un tel arc rajouté à l'arborescence crée un circuit alors FIN2;

Si un tel arc rajouté à l'arborescence crée un cycle qui n'est pas un circuit, aller en D) ;

Fin;

D) Modification de l'arborescence :

Début

Enlever de A l'arc marqué d'extrémité terminale j et ajouter à A l'arc(i,j);

Pour j et tous ses descendants dans A diminuer les  $\lambda$  de la quantité  $\delta = \lambda_j - \lambda_i - v(i,j)$ ;

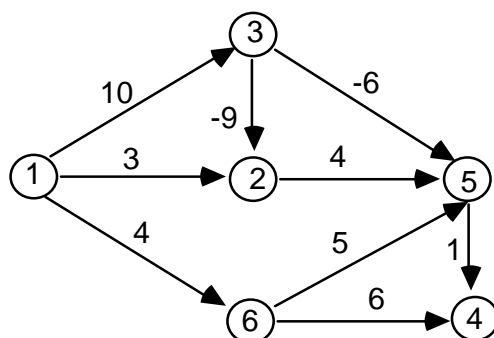
Aller en C);

Fin;

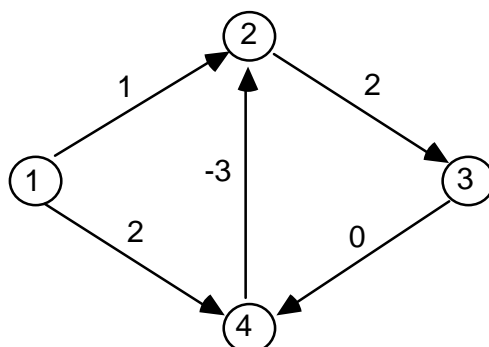
Fin.

QUESTIONS

1) Appliquer l'algorithme au graphe  $G_1$  ci-dessous.



2) Appliquer l'algorithme au graphe  $G_2$  ci-dessous.



3) On rappelle qu'une arborescence de racine 1 peut être défini par une fonction père de  $X - \{1\} \rightarrow X$ , telle que 1 soit ascendant de tous les sommets. On admettra qu'à la fin de A) : tous les sommets ont un potentiel fini, et que l'ensemble des arcs ayant permis de fixer le potentiel, donc satisfaisant  $\lambda_j - \lambda_i = v(i,j)$ , définissent une arborescence de racine 1. Montrer que l'arborescence A a la propriété suivante : " $\lambda_i$  est la valeur du chemin de l'arborescence A allant de 1 à i".

4) Expliciter FIN2.

5) Montrer qu'à chaque fois qu'on passe par D), on redéfinit une nouvelle arborescence de racine 1 ayant les mêmes propriétés.

6) Montrer qu'en l'absence de circuit absorbant, l'algorithme se termine en FIN1 après un nombre fini d'étapes et qu'alors  $\lambda_i$  est la valeur minimale d'un chemin allant de 1 à i. En déduire sa validité. Est-il polynomial ? Et s'il y avait des circuits absorbants ?



Exercice 1 : ORDONNANCEMENT

Un projet comporte sept tâches a,b,c,d,e,f,g. Pour chaque tâche, le tableau ci-dessous rapporte sa durée et les tâches préalables.

- 1) Dessiner le graphe potentiel-tâches et calculer les dates au plus tôt, les dates au plus tard et le chemin critique.
- 2) Dessiner un graphe PERT simplifié avec un minimum d'arcs fictifs.

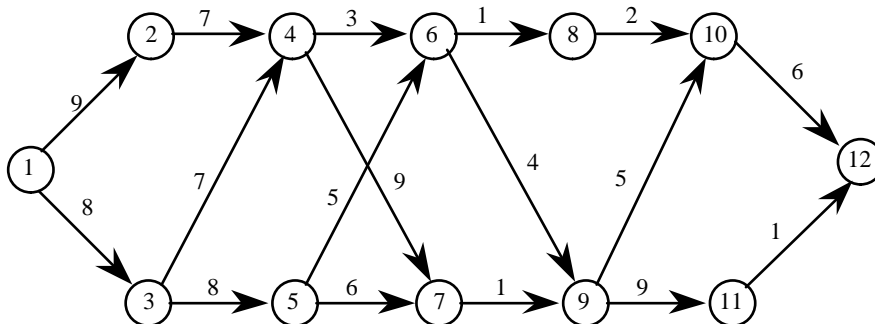
tâches	durées	tâches préalables
a	4	
b	3	
c	6	a et b
d	2	b
e	8	
f	7	c, d et e
g	1	c

Exercice 2 : L'itinéraire de Michel STROGOFF (d'après ROSEAUX)

Partant de MOSCOU, Michel STROGOFF, courrier du tsar, devait rejoindre IRKOUTSK. Avant de partir, il avait consulté une voyante qui lui avait dit entre autres choses:

“après KAZAN méfiez-vous du ciel, à OMSK attention aux tartares, dans TOMSK attention aux yeux, après TOMSK méfiez-vous de l'eau et, surtout prenez garde partout à un grand brun avec des bottes noires”.

STROGOFF avait alors reporté sur une carte pour chaque liaison entre deux villes ses “chances” de réussite : ces chances étaient exprimées par un nombre compris entre 1 et 10 (mesurant le nombre de chances sur 10 de succès). Ignorant le calcul des probabilités, il avait alors choisi son itinéraire en maximisant la somme globale des chances.



Les numéros des villes sont les suivants: MOSCOU(1), KAZAN(2), PENZA(3), PERM(4), OUF(5), TOBOLSK(6), NOVO-SAIMSK(7), TARA(8), OMSK(9), TOMSK(10), SEMIPALATINSK(11), IRKOUTSK(12).

1. Déterminer l'itinéraire de Michel STROGOFF.

2. Quelle était alors sous l'hypothèse d'indépendance des variables aléatoires "chances de succès", la probabilité pour que STROGOFF réussisse?

3. Quel aurait été son itinéraire s'il avait connu les principes du calcul des probabilités?

### Exercice 3 : UN PROBLEME D'INVESTISSEMENT

1) Un exemple. On dispose de 6 millions à investir dans 3 régions.

Le tableau suivant donne les bénéfices résultant des sommes investies.

Déterminer la politique d'investissement optimale entre les trois régions à l'aide d'une méthode de "programmation dynamique". L'idée est d'associer à la donnée un graphe en niveaux. Le niveau 0 ne contient que le sommet (0,0), (car aucune somme n'a encore été investie). Le niveau 1 contient les sommets (1,0), (1,1), (1,2), (1,3), (1,4), qui correspondent aux sommes cumulés investies dans la région 1. Le niveau  $i$  contient les sommets (i,0), (i,1), (i,2), (i,3), (i,4), (i,5), (i,6), qui correspondent aux sommes investies dans les régions 1..i, ( $i = 2, 3$ ). On introduit les arcs entre les niveaux  $i$  et  $i+1$ , valués par les sommes investies dans la région  $i+1$ . Le dernier sommet est (3,6). Il s'agit de chercher un chemin de valeur maximale dans ce graphe.

	Région I	Région II	Région III
1 million	0,2	0,1	0,4
2 millions	0,5	0,2	0,5
3 millions	0,9	0,8	0,6
4 millions	1	1	0,7

2) Le cas général. Plus généralement on dispose de  $B$  millions à investir dans  $n$  régions. On pose  $f_i(y)$  le bénéfice optimal pour un investissement cumulé d'une somme  $y$  dans les régions 1, 2, ...,  $i$ . On a  $f_0(0) = 0$ . Donner une formule de récurrence reliant  $f_i$  à  $f_{i-1}$  pour  $i$  de 1 à  $n$ .

Quelle est en fonction de  $n$  et de  $B$  la complexité de la méthode de programmation dynamique et la complexité d'énumération de toutes les solutions possibles.

LE PROBLEME DU SAC A DOS (d'après ROSEAUX)

Un randonneur, partant pour une longue excursion, détermine avec soin le contenu de son sac à dos. Compte tenu des équipements indispensables déjà chargés, le poids total de nourriture emportée ne devra pas excéder 16 kilogrammes. Il dispose, en quantités limitées, de trois types d'aliments, de valeurs nutritives variables, dont les poids unitaires sont différents. Les aliments sont conditionnés par unités non fractionnables.

Aliments	I	II	III
poids unitaires en kilo	7	5	2
quantités disponibles	4	3	4
valeurs nutritives	15	10	4

Le randonneur choisit la quantité de chaque aliment à emporter, de façon à maximiser la valeur nutritive totale, tout en tenant compte de la limite de 16 Kg qu'il s'est fixée.

Il va utiliser la "programmation dynamique" pour résoudre ce problème dit du "Knapsack" ou sac à dos. D'une manière générale, soit :

$n$ , le nombre d'aliments différents;

$p_i$  le poids unitaire de l'aliment  $i$  ( $i \in [1, n]$ );

$q_i$  le nombre d'unités disponibles de l'aliment  $i$  ( $i \in [1, n]$ );

$c_i$  la valeur nutritive de l'aliment  $i$  ( $i \in [1, n]$ );

$x_i$  la quantité de l'aliment  $i$  emportée en nombre d'unités ( $i \in [1, n]$ ) (ce sont les inconnues);

et  $P$  le poids maximal de nourriture qu'il peut emporter.

1) Donner une formalisation du problème en tant que programme linéaire en nombre entiers.

2) En notant  $f_i(Q)$  la valeur nutritive maximale du sac à dos chargé optimalement avec uniquement les  $i$  premiers produits, et de poids total  $Q$  ( $Q$  inférieur à  $P$ ). Donner les formules d'optimisation séquentielle, c'est-à-dire des formules reliant la fonction  $f_i$  à la fonction  $f_{i-1}$ .

3) Tracer le graphe des décisions associé au problème, construit de façon analogue à l'exercice de TD précédent sur le problème de l'investissement.

4) Résoudre numériquement.

CONSTRUCTION D'UNE MAISON (d'après ROSEAUX) :

Dans la construction d'une maison, il faut commander et livrer le matériel électrique avant de poser puis d'inspecter les câblages. De même, dès que le matériel nécessaire aux travaux de plomberie a été livré (après avoir été commandé), on peut effectuer les travaux de plomberie qui se font en deux étapes "à l'extérieur" puis "à l'intérieur", car nous n'avons qu'un seul plombier.

Il faut commander spécialement les parements extérieurs sablés, servant au briquetage extérieur et, de toute façon, on ne peut poser ces briques tant que l'ossature de la maison n'a pas été édifiée. La première étape dans la construction de la maison est constituée par les travaux de terrassement, après quoi, on peut poser les fondations et commencer à construire l'ossature. Les travaux de

plomberie extérieure peuvent être commencés quand les fondations sont terminées et ceux de plomberie intérieure une fois que l'ossature est terminée.

Quand l'ossature est terminée, on peut construire la charpente du toit et commencer à poser les câblages. Les revêtements intérieurs doivent attendre que la charpente soit terminée, les câblages inspectés et les travaux de plomberie intérieure achevés. Après les revêtements intérieurs, on effectue les aménagements intérieurs, c'est-à-dire poser les planchers, faire la décoration, mais cela, bien sûr après que la couverture ait été posée. Ces tuiles doivent être commandées spécialement.

Une fois que la couverture, les revêtements intérieurs et le briquetage sont terminés, on nettoie l'extérieur, puis on passe aux aménagements intérieurs. Et une fois que les aménagements intérieurs sont terminés, on effectue une inspection générale. La maison est achevée lorsque l'inspection générale et les extérieurs sont terminés.

Question 1 : Enumérer les différentes tâches, puis tracer le tableau des tâches préalables.

Question 2 : Dessiner le graphe potentiel-tâches. Calculer l'ordonnancement au plus tôt en utilisant l'algorithme de BELLMAN. On utilisera le tableau décrivant le problème. On pourra dresser le tableau des prédécesseurs. Quels sont le ou les chemins critiques ?

Question 3 : Tracer un graphe PERT aussi simple que possible.

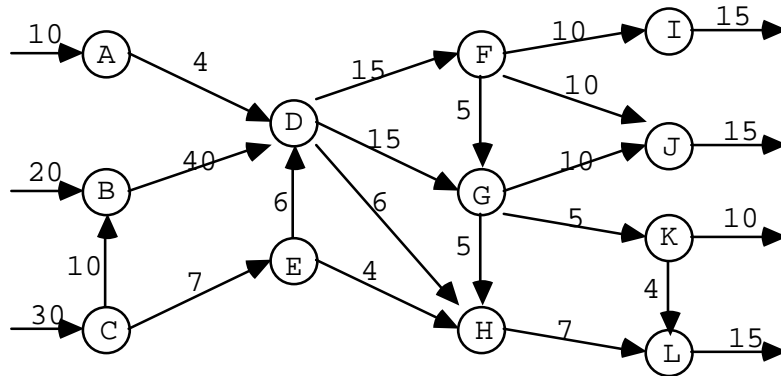
Question 4 : Calculer les dates au plus tôt et au plus tard des événements introduits dans ce graphe PERT. En déduire les marges totales de chacune des tâches.

Voici le tableau des tâches avec leurs durées et les tâches préalables.

	Durée en semaines	tâches préalables
A. commande et livraison des câblages	3	Aucune
B. pose des câblages	4	A, I
C. inspection des câblages	1	B
D. commande et livraison du matériel de plomberie	4	Aucune
E. travaux de plomberie extérieure	2	D, H
F. travaux de plomberie intérieure	5	E, I
G. terrassement	1	Aucune
H. fondations	3	G
I. construction de l'ossature	5	H
J. commande et livraison des briques sablées	6	Aucune
K. briquetage	3	I, J
L. commande et livraison des tuiles	14	Aucune
M. construction de la charpente	2	I
N. pose de la couverture	2	M, L
O. revêtements intérieurs	3	M, F, C
P. aménagements intérieurs	3	O, N
Q. inspection générale	2	P
R. nettoyage extérieur	1	N, K, O
S. aménagements extérieurs	3	R

PROBLEME DE CIRCULATION URBAINE :

Le réseau routier que doit entretenir la ville de Fulkerson City est schématisé ci-dessous (les capacités des arcs représentent le débit maximal en voitures par heure).



1) Soit N le nombre maximum de voitures pouvant circuler par heure à travers ce réseau .

1.1) Calculer N .

1.2) Caractériser les voies que l'on peut créer de façon à augmenter N. De combien peut-on augmenter N ?

2) Les édiles locaux souhaitent améliorer la circulation en augmentant le nombre N grâce à la création de nouvelles voies . Dans la zone urbaine, seuls sont matériellement possibles la création d'une voie HK et l'élargissement de la rue EG jusqu'alors réservée aux piétons. Le choix devra donc se faire entre :

- a) créer HK;
- b) élargir EG;
- c) créer HK et élargir EG.

2.1) Si la politique a) est choisie, quelle capacité maximum doit-on prévoir pour la voie HK ? Dans ce cas, de combien augmente N ?

2.2) Si la politique b) est choisie, quelle autre voie est-on amené à créer si l'on veut augmenter N ? Donner les capacités des deux nouvelles voies et la nouvelle valeur de N .

2.3) Evaluer N dans le cas c) .

2) PROBLEME : SECOND PLUS COURT:

Le but de ce problème est de déterminer un second plus court chemin entre les sommets 0 et n-1 d'un graphe  $G=(X,U,v)$  positivement valué et sans circuit . Soit  $\gamma(0,n-1)$  un plus court chemin de 0 à n-1 . Un second plus court chemin de 0 à n-1 est un chemin de valeur minimale parmi l'ensemble des chemins de 0 à n-1 qui diffèrent de  $\gamma(0,n-1)$  par au moins un sommet. Un tel chemin existe s'il y a , au moins , deux chemins de 0 à n-1 dans G , ce que nous supposons dans la suite .

On se propose de montrer que **second** , fourni par l'algorithme suivant , définit un second plus court chemin de 0 à n-1 .

Algorithme second plus court :

début

1) Appliquer l'algorithme de Dijkstra pour obtenir l'arborescence A des plus courts chemins allant de 0 à i et les potentiels  $\lambda(i)$  de 0 à i pour tous les sommets i de G .

(N.B.: On notera  $\gamma(r,s)$  le chemin , s'il existe , de r à s dans A )

2) Déterminer  $\gamma(0,n-1)=(y_0=0,y_1,\dots,y_p=n-1)$  .

3) Poser valeur:= $+\infty$  ;

Pour j:=1 jusqu'à p faire

pour tout  $k \in ( U^-(y_j) - \{y_{j-1}\} )$  faire

début

$\alpha := \lambda(k) + v(k,y_j) + ( \lambda(n-1) - \lambda(y_j) )$  ;

si  $\alpha < \text{valeur}$  alors

début

valeur :=  $\alpha$  ; pivot1 := k ; pivot2 :=  $y_j$  ;

fin;

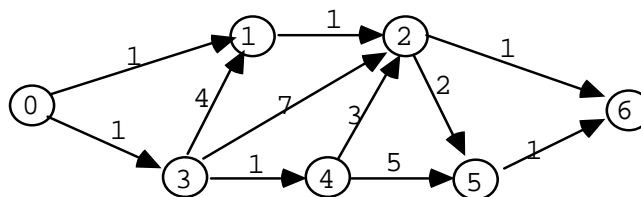
fin;

4) **Second** :=  $\gamma(0,\text{pivot}_1) + (\text{pivot}_1,\text{pivot}_2) + \gamma(\text{pivot}_2,n-1)$  .

fin.

### QUESTIONS :

1) Appliquer l'algorithme au graphe G suivant :



N.B. : On rapportera successivement:

- le tableau de Dijkstra présenté en cours;
- l'arborescence A;
- le chemin  $\gamma(0,n-1)$ ;
- les différentes valeurs de  $\alpha$ ;
- **second**.

Vérifier que **second** est bien un second plus court chemin dans G .

2) Etude de la complexité en fonction de n et m:

On suppose que le graphe est codé par la file des successeurs et des prédécesseurs.

2.1) Quelles sont les complexités des phases 1 , 2 , 3 et 4 de l'algorithme ?

En déduire la complexité totale .

2.2) Quelle amélioration peut-on proposer pour diminuer cette complexité ?

3) Preuve de l'algorithme :

On note **premier** =  $\gamma(0,n-1) = (y_0=0,y_1,\dots,y_p=n-1)$  le plus court chemin obtenu dans la phase 1) de l'algorithme et **deuxième** =  $(z_0=0,z_1,\dots,z_q=n-1)$  un second plus court chemin de 0 à n-1 .

3.1) Montrer qu'il existe un entier r tel que  $y_p=z_q$  ,  $y_{p-1}=z_{q-1}$  , ...,  $y_{p-r}=z_{q-r}$  et  $y_{p-r-1}\neq z_{q-r-1}$ .

3.2) Quelle est la propriété remarquable du chemin  $(z_0=0,z_1,\dots,z_{q-r-1})$  ?

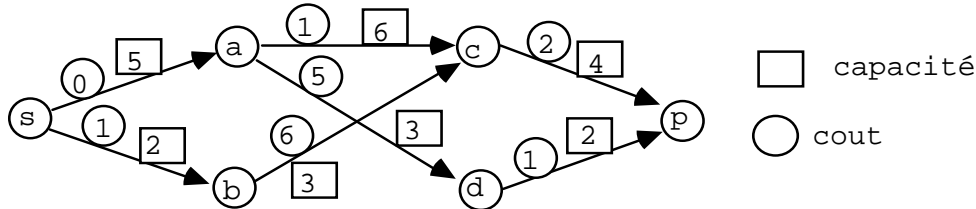
3.3) En déduire la validité de l'algorithme.





1) EXERCICE : Algorithme de ROY:

Appliquer l'algorithme de Roy pour déterminer un flot maximal de coût minimal dans le graphe suivant.



On dessinera les graphes d'écart intermédiaires et on précisera les chaînes améliorantes successives utilisées. Calculer le coût du flot obtenu.

PROBLEME D'AFFECTION :

On veut affecter six programmes a,b,c,d,e,f à six programmeurs A,B,C,D,E,F en minimisant la somme des temps de travail et de façon à ce qu'un programmeur fasse exactement un programme. Le tableau ci-dessous rapporte les différents temps de travail. Pour résoudre ce problème d'affectation, on va appliquer l'algorithme hongrois qui permet plus généralement d'affecter n individus à n travaux en minimisant la somme des coûts d'affectation.

	a	b	c	d	e	f
A	14	6	18	16	63	15
B	41	78	44	73	70	25
C	44	81	36	80	80	78
D	46	74	5	25	83	3
E	72	32	55	51	3	81
F	69	76	12	99	83	30

1) Montrer qu'on ne change pas la solution du problème en soustrayant ou ajoutant une valeur numérique  $\delta$  à tous les éléments d'une rangée (ligne ou colonne). Seule la valeur de la solution est modifiée.

Le faire sur l'exemple en soustrayant le plus petit élément de chaque rangée en commençant par les lignes et en terminant par les colonnes, (l'intérêt est de faire apparaître des zéros).

2) On associe au nouveau tableau un graphe biparti reliant les programmeurs aux programmes quand le coût réduit (obtenu après soustraction des minima en lignes et en colonnes) vaut 0. Les arcs du graphe biparti sont dits admissibles.

On introduit aussi un réseau de transport en reliant la source s aux programmeurs par des arcs de capacité 1, le puits p aux programmes par des arcs de capacité 1. Les arcs admissibles sont de capacité infinie.

a) Construire le réseau de transport.

- b) Calculer un flot maximal sur ce réseau et en déduire un couplage maximal de coût nul.
- c) Le marquage de Ford-Fulkerson à l'optimum permet de marquer les lignes  $X^*$  de  $X$  et les colonnes  $Y^*$  de  $Y$ .

Vérifier sur l'exemple qu'il n'y a pas d'arc admissible entre  $X^*$  et  $Y-Y^*$ .

Vérifier sur l'exemple que les arcs entre  $X-X^*$  et  $Y^*$  ont un flux nul.

Justifier les deux propriétés.

d) Soit  $\delta$  le plus petit coût réduit d'affectation d'un programmeur de  $X^*$  à un programme de  $Y-Y^*$ . Calculer  $\delta$ . Ajouter  $\delta$  aux colonnes de  $Y^*$  et soustraire  $\delta$  aux lignes de  $X^*$ . Pourquoi ces opérations sont-elles licites ?

e) On obtient une nouvelle matrice de coûts. Vérifier que les arcs de flux non nuls restent admissibles. Quelle est la propriété des arcs qui étaient admissibles et qui ne le sont plus ?

Quelle est la propriété des arcs devenus admissibles? Quelle est la capacité de l'ancienne coupe minimale ? Montrer que le nouveau marquage contient le précédent ?

f) Calculer le nouveau flot maximal ?

g) Vérifier que la procédure pratique suivante permet de calculer la nouvelle matrice.

#### Procédure pratique.

-rayer toute ligne non marquée ;

-rayer toute colonne marquée ;

-soit  $\delta$  le plus petit élément non rayé ;

-retrancher  $\delta$  aux éléments non rayés ;

-ajouter  $\delta$  aux éléments rayés deux fois.

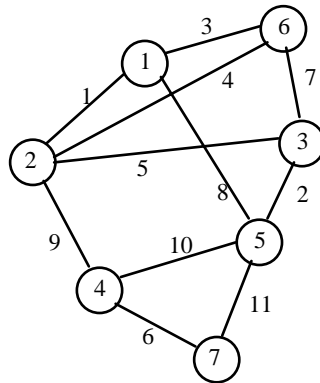
h) On itère autant de fois que nécessaire la procédure pratique et le calcul du flot :

-le faire;

-à quelle condition s'arrête-t-on ?

-en déduire l'optimalité de la méthode.

PROBLEME: ALGORITHME DE KRUSKAL:



L'algorithme de KRUSKAL permet de déterminer un arbre couvrant de coût minimal.

**On supposera que les coûts sont tous distincts.**

KRUSKAL

{ par convention l'arête  $e_L$  s'écrira  $e_L = [I, J]$  avec I strictement plus petit que J }

Début

Indicer les arêtes en ordre de coûts croissants:  $v(e_1) \leq v(e_2) \leq \dots \leq v(e_m)$

Pour H:=1,N faire CONNEXE(H):=H; K:=1; {le graphe partiel est initialement vide}

  Pour L:=1,M faire

    Début

      soit  $e_L = [I, J]$ ;

      Si (CONNEXE(I) différent de CONNEXE(J)) alors

        Début

          AUXI=CONNEXE(J);  $f_K = [I, J]$ ; {  $f_K$  est mis dans le graphe partiel }; K:=K+1;

          Pour H:=1,N faire

            Si (CONNEXE(H)=AUXI) alors CONNEXE(H):=CONNEXE(I)

        Fin

    Fin

Fin

Question 1: Faire tourner l'algorithme sur le graphe ci-dessus: on dessinera les graphes partiels successifs et les onze valeurs successives du tableau CONNEXE obtenu avant chaque itération de la boucle externe.

Question 2: Evaluer en fonction de N et M la complexité de l'algorithme. On pourra distinguer deux types d'itérations dans la grande boucle.

Question 3 (cette question est indépendante de l'algorithme)

On rappelle que l'on est dans le cas où tous les coûts sont distincts.

Montrer qu'un arbre couvrant est de coût minimal si et seulement l'intersection de chaque cocycle de G et de l'arbre contient l'arête de coût minimal du cocycle de G.

Question 4: Montrer que l'algorithme détermine l'arbre couvrant de coût minimal.

## PROGRAMMES DE TRANSPORT.

On dispose de  $a_1, a_2, \dots, a_n$  unités dans des dépôts  $1, 2, \dots, n$ . On veut transporter ces unités vers des destinations  $1, 2, \dots, m$  où se manifestent des demandes  $b_1, b_2, \dots, b_m$ , avec :  $\sum_{i=1}^n a_i = \sum_{j=1}^m b_j$

La méthode appliquée au problème d'affectation se généralise. On fait apparaître des zéros en lignes et en colonnes. On introduit la notion d'arcs admissibles et on trace un réseau de transport sur lequel on recherche un flot maximum. On calcule ensuite un  $\delta$  pour modifier ce réseau de façon à finalement obtenir, après un certain nombre d'itérations, un flot satisfaisant toutes les demandes et qui est de coût minimal.

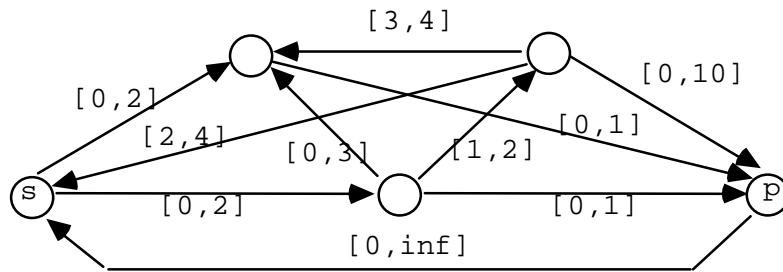
On demande d'appliquer cette méthode au problème suivant. On a trois dépôts où sont disponibles respectivement 3, 4, 7 hélicoptères et on doit répartir ces hélicoptères sur quatre sites où se manifestent respectivement des demandes de 1, 3, 5, 5 hélicoptères. Les coûts unitaires de transport sont donnés dans le tableau suivant. Déterminer la solution optimale.

2	4	3	2	4
6	2	6	4	3
5	6	7	7	7
1	3	5	5	

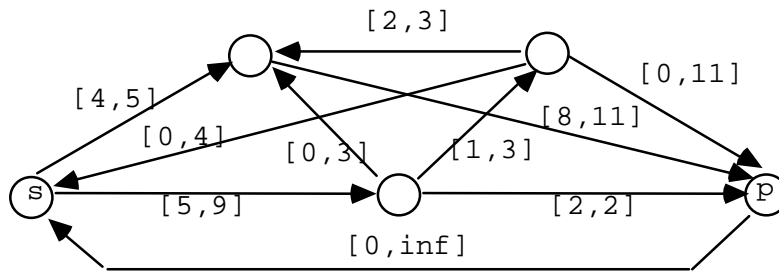
**PROBLEMES DE FLOTS**

Exercice 1 : flot canalisé

Construire un flot de valeur maximale sur l'arc ps, lorsqu'il existe, dans les réseaux suivants (les capacités min et max sont indiquées par  $[b_{ij}, c_{ij}]$ ).



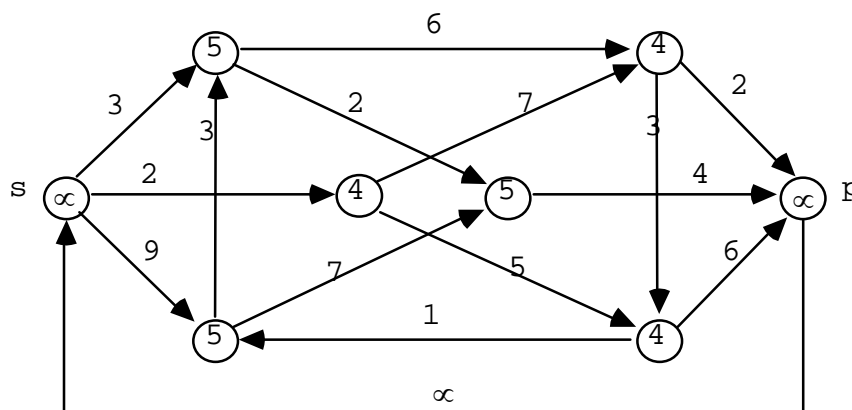
Réseau n°1



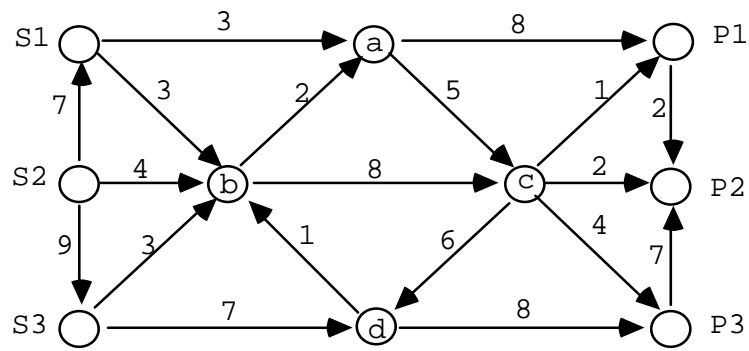
Réseau n°2

Exercice 2 : flot maximal

Considérons le réseau suivant dont, non seulement les arcs sont munis de capacité, mais aussi les sommets. Trouver un flot de valeur maximale sur ce réseau en expliquant votre méthode.



Exercice 3 : satisfaction de demandes



Dans ce réseau, les sommets S1, S2, S3 sont des dépôts qui disposent respectivement des quantités 5 , 10 , 5 d'un même produit . Les sommets P1, P2, P3 sont des clients dont les demandes sont respectivement : 5, 10, 5. Ces demandes peuvent-elles toutes être satisfaites à travers ce réseau ?