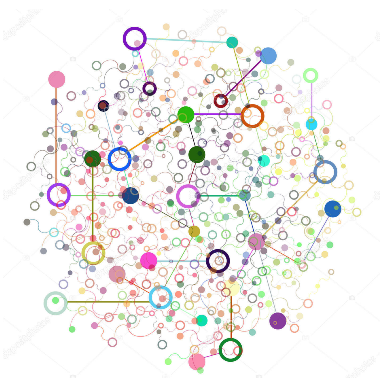
République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Centre Universitaire Ahmed Zabana de Relizane
Institut des Sciences et de la Technologie
Département d'Informatique

Théorie des Graphes

BENOTMANE ZINEB

2^{e} année Informatique
(LMD)



Théorie des graphes

BENOTMANE Zineb*

2 février 2019

Table des matières

*Centre Universitaire Ahmed Zabana de Relizane

Table des figures

Résumé

Un exemple de fichier LaTeX de quelques lignes.

Première partie

Cours

1 Chapitre 1 : Notions de graphes

Les graphes permettent de représenter graphiquement des objets et leurs relations.
Un graphe est un ensemble de points (appelés sommets), muni d'une relation binaire dont les couples sont appelés arcs ou arêtes.
La théorie des graphes représente les techniques et outils mathématiques et informatique qui permettent de démontrer des propriétés, d'en déduire des méthodes de résolution de problèmes, des algorithmes sur les graphes, etc, par exemple : pour aller d'un endroit X à un endroit Y, quel est le chemin le plus rapide ?

1.1 Formulation

Un graphe noté $G = (X, U)$ est représenté par :
- Un ensemble$X$= {x1, x2, . . . , xn} dont les éléments sont appelés sommets ($|X| = n$).
- Un ensemble U = {u1, u2, . . . , um} dont les éléments sont appelés arêtes ou arcs ($|U| = m$).

1.2 Graphes non orientés

Un graphe non orienté est composé d'un ensemble de sommets et d'un ensemble d'arête. L'exemple de la figure 1 est un graphe non orienté comportant 5 sommets et 7 arêtes. Dans le graphe G de la figure 1 :

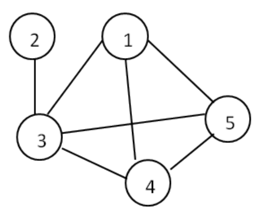


FIGURE 1 – Graphe non orienté (G)

X={1,2,3,4,5}
U = {(1,3) ; (1,4) ; (1,5) ; (2,3) ; (3,4) ; (3,5) ; (4,5)}

1.3 Graphe orienté

Un graphe orienté est un graphe dont les arêtes sont orientées (fléchées). On distingue alors le sommet origine de l'arête et son extrémité. Dans un graphe orienté, les éléments de U sont appelés arcs.

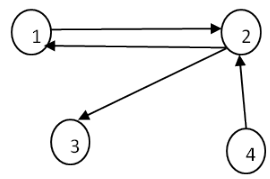


FIGURE 2 – Graphe orienté (G)

Dans le graphe G de la figure 2 :
X={1,2,3,4}
U={(1,2) ;(2,3) ;(4,2) ;(2,1)}

1.4 Quelques notions

- Une arête u est une paire de sommets (x_i, x_j).
- Les sommets x_i et x_j sont les extrémités de l'arête.
- Deux sommets reliés par au moins une arête sont dits adjacents ou voisins.
- Une arête partant et arrivant au même sommet est appelée boucle.
- L'ordre d'un graphe est le nombre de sommets de ce graphe.
- Deux arêtes d'un graphe sont adjacentes si elles ont au moins un sommet en commun.
- Dans un graphe, le degré de chaque sommet est le nombre d'arêtes dont il est l'une des extrémités Attention ! Il ne faut pas oublier de compter deux fois les boucles, car le sommet est deux fois l'extrémité de cette arête.
- L'ensemble des successeurs d'un sommet x_i est l'ensemble $succ(x_i)$ notés $\Gamma^+(x_i)$. De manière analogue, l'ensemble des prédécesseurs d'un sommet x_i est l'ensemble $pred(x_i)$ noté $\Gamma^-(x_i)$. Et dans ce cas l'ensemble des voisins de x_i est simplement l'ensemble $succ(x_i) \cup pred(x_i)$.
- Arcs incidents à x_i : entrants $\Gamma^-(x_i)$ ou sortants $\Gamma^+(x_i)$.
- Le nombre d'arêtes incidentes à un sommet est le nombre d'arêtes sortant ou rentrant du sommet.
- Le degré d'un sommet $x_i$de G est le nombre d'arêtes incidentes à x_i. Il est noté $d(x_i)$.
- Lorsque $d(x_i) = 0$, on dit que le sommet xi est *isolé*.
- Lorsque $d(x_i) = 1$, on dit que le sommet xi est *pendant*.
- Lorsque plusieurs arêtes relient deux sommets, on les appelle des *arêtes multiples*.

1.5 Représentation des graphes

1.5.1 Table des Successeurs et Prédécesseurs

Un graphe peut être représenté par ses successeurs et prédécesseur ; il s'agit d'une table à simple entrée ou chaque ligne correspond à un sommet et comporte la liste des successeurs ou des prédécesseurs de ce sommet. Soit l'exemple de la figure 3.

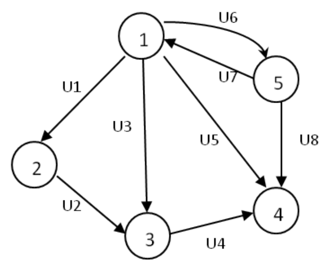


FIGURE 3 – Exemple d'un graphe orienté(G)

La table des successeurs correspondante au graphe G de la figure 3 est représentée dans le tableau (a) de la figure 4; la table des prédécesseurs est représentée dans le tableau (b) de la figure 4.

x_i	$\omega^+(x_i)$
1	2, 3, 4, 5
2	3
3	4
4	/
5	1, 4

(a)

x_i	$\omega^-(x_i)$
1	5
2	1
3	1, 2
4	1, 3, 5
5	1

(b)

FIGURE 4 – Tables des successeurs/ prédécesseurs

1.5.2 Matrice d'adjacence

Les outils classiques d'algèbre linaire peuvent également être utilises pour coder les graphes. La première idée consiste à considérer chaque arc comme un lien entre deux sommets. Considérons un graphe $G = (X, U)$ comportant n sommets. La matrice d'adjacence de G est égale à la matrice $A = (a_{ij})$ de dimension $n * n$ telle que :

$$a_{ij} = \begin{cases} 1 & \text{si } \{x_i, x_j\} \in U \\ 0 & \text{sinon.} \end{cases} \quad (1)$$

Une telle matrice, ne contenant que des 0 et des 1 est appelée, de manière générale, une matrice booléenne.

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix} \quad (2)$$

Dans la matrice représentée dans la formule 2, les lignes représentent les sommets du départ tandis que les colonnes représentent les sommets d'arrivée.
Un graphe orienté quelconque a une matrice d'adjacence quelconque, alors qu'un graphe non orienté possède une matrice d'adjacence symétrique. L'absence de boucle se traduit par une diagonale nulle.

1.5.3 Matrice d'incidence

Considérant un graphe orienté sans boucle $G = (X, U)$ comportant n sommets $x_1, ..., x_n$ et m arcs $u_1, ..., u_m$. On appelle matrice d'incidence (aux arcs) de G la matrice $M = (m_{ij})$ de dimension $n * m$ telle que :

$$m_{ij} = \begin{cases} 1 & \text{si } x_i \text{ est l'extrémité initiale de } u_j \\ -1 & \text{si } x_i \text{ est l'extrémité terminale de } u_j \\ 0 & \text{sinon.} \end{cases} \tag{3}$$

Dans la matrice représentée dans la formule 4, les lignes représentent les sommets tandis que les colonnes représentent les arcs.

$$M = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 1 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & -1 & 1 & 1 \end{pmatrix} \tag{4}$$

Pour un graphe non orienté sans boucle, la matrice d'incidence (aux arêtes) est définie par :

$$m_{ij} = \begin{cases} 1 & \text{si } x_i \text{ est l'extrémité de } u_j \\ 0 & \text{sinon.} \end{cases} \tag{5}$$

1.6 Types de graphes

1.6.1 Graphe simple

Un graphe est simple s'il ne contient ni boucle ni arêtes multiples (figure 5).

1.6.2 Multi-graphe

Un multi-graphe est un graphe qui n'est pas simple (figure 5).

1.6.3 Graphe complet

Un graphe complet est un graphe où chaque sommet est relié à tous les autres. Le graphe complet d'ordre n est noté K_n. Dans ce graphe chaque sommet est de degré $n - 1$ (figure 6).

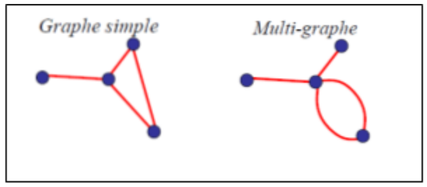


FIGURE 5 – Graphe simple - Multigraphe

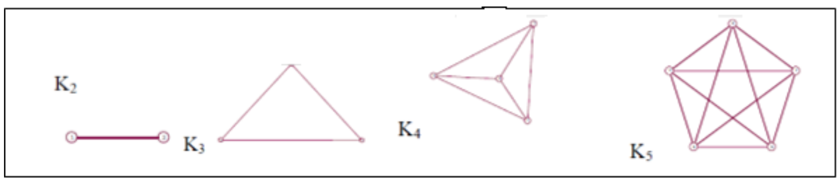


FIGURE 6 – Exemple d'un graphe complet

1.6.4 Graphe Biparti

Un graphe $G(X, U)$ est biparti ou bipartite si l'ensemble de ses sommets X peut être divisé en deux ensembles A et B, de sorte que :

- Les éléments de A ne sont reliés entre eux par aucune arête.
- Les éléments de B ne sont reliés entre eux par aucune arête.

En d'autres termes, les sommets du même sous ensemble ne peuvent pas être reliés par une arrête : $X = A \cup B$ et $A \cap B$.
La figure 7 est un exemple d'un graphe biparti.

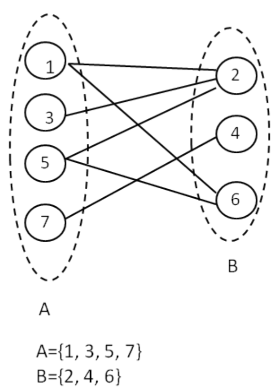


FIGURE 7 – Graphe Biparti

Les arêtes relient uniquement des éléments de A à des éléments de B. Un graphe bipartite complet $G(X, U)$, noté Kp, q, est un graphe bipartite où chaque sommet de A est relié à tous les sommets de B par une arête (voir figure 8).

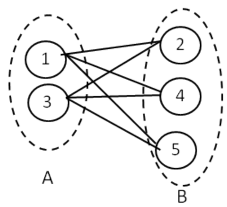


FIGURE 8 – Graphe Biparti complet

1.6.5 Graphe partiel

Soit $G = (X, U)$ un graphe. Le graphe $\grave{G} = (X, \grave{U})$ est un graphe partiel de G, si $\grave{U}$ est inclus dans U. Autrement dit, on obtient $\grave{G}$ en enlevant une ou plusieurs arêtes au graphe G.

1.6.6 Sous graphe

Un sous-graphe de G est un graphe $H = (V, U(V))$ tel que V est un sous ensemble de X, et $U(V)$ sont les arêtes induites par U sur V, c'est-à-dire les arêtes de U dont les deux extrémités sont des sommets de V.
La figure 9 est un exemple illustrant un graphe g, son graphe partiel et son sous graphe.

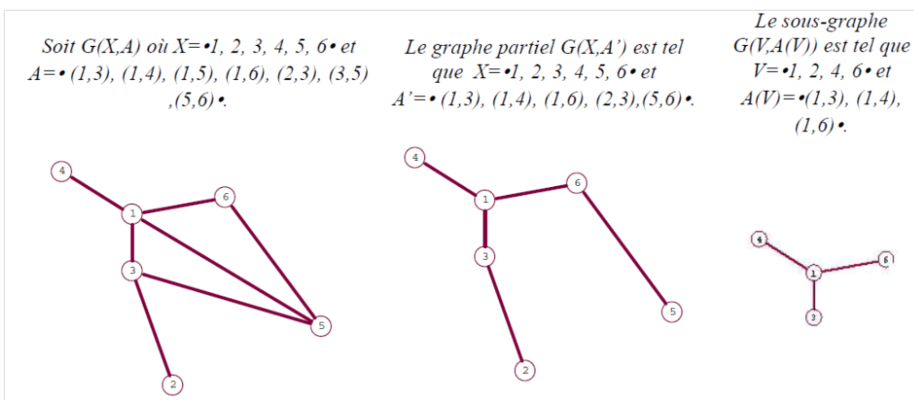


FIGURE 9 – Graphe partiel- Sous graphe

1.6.7 Graphe planaire

Un graphe planaire est un graphe qui a la particularité de pouvoir se représenter dans un plan sans qu'aucune arête, courbe ou rectiligne, n'en croise une

autre.
Si on arrive à dessiner le graphe sans qu'aucune arête n'en coupe une autre (les arêtes ne sont pas forcément rectilignes), on dit que le graphe est planaire (figure 10). On observe qu'un graphe planaire découpe le plan en plusieurs **ré-**

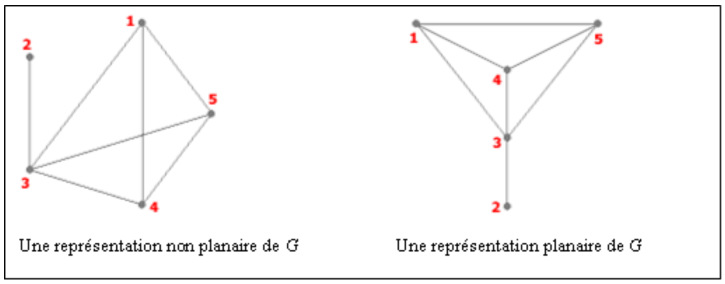


FIGURE 10 – Représentation planaire

gions. Une **face** d'un graphe est par définition une région du plan limitée par des arêtes et qui ne contient ni sommets ni arêtes dans son intérieur.
Deux faces r et s sont dites adjacentes si leurs contours ont au moins une arête commune ; deux faces qui ne se touchent que par un sommet ne sont pas adjacentes.
Le **degré** d'une face F (ou région) noté $d(F)$ est égal à la longueur du cycle ou de la chaîne fermée qui limite F.

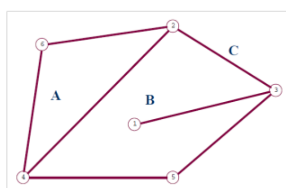


FIGURE 11 – Exemple d'un graphe planaire

Soit le graphe planaire de la figure 11, il 6 sommets, 7 arêtes, divise le plan en 3 faces A, B et C. On remarque que les faces A et B sont limitées, alors que la face C, extérieure, est illimitée. $d(A) = 3; d(B) = 6; d(C) = 5$.
La Formule d'Euler Dans un graphe planaire à S sommets, F faces et A arêtes, on a : $S + F = A + 2$.

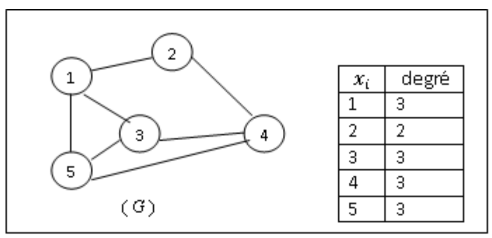
1.7 Algorithme de K-coloration d'un graphe

On appelle **k-coloration** de $G = (X, E)$ une partition de l'ensemble des sommets X de G en **k-classes** $(X_1, X_2, ..., X_k)$ de telle façon que :
- Deux sommets d'une même classe ne soient pas adjacents ;
- Les sommets d'une classe sont coloriés de la même couleur.

La méthode utilisée pour avoir la k-coloration d'un graphe est la suivante :
On commence à établir une liste ordonnée de sommets (suivant l'ordre décroissant de leur degré).
Tant qu'il reste des sommets à colorier, exécuter les actions suivantes :

1. Choisir une nouvelle couleur appelée *couleur d'usage* ;
2. Chercher dans la liste de sommets le premier sommet non coloré et le colorer avec la couleur d'usage ;
3. Examiner tour à tour, dans l'ordre de la liste, tous les sommets non colorié et colorer chaque sommet non adjacent à un sommet déjà coloré avec la couleur d'usage.

On appelle le nombre chromatique $\sigma(G)$ de G, le nombre minimum pour lequel le graphe G est k-coloriable.
si $\sigma(G) = 2$, alors G est bi-partie.
Soit l'exemple de la figure 12. Le tableau de la figure représente les degrés de chaque sommet du graphe G. Nous disposons de la liste initiale ordonnée :



x_i	degré
1	3
2	2
3	3
4	3
5	3

Figure 12 – Exemple de l'application de la méthode de k-coloration

$1 - 3 - 4 - 5 - 2$
On choisit une couleur du départ, soit la couleur *rouge*, on commence par colorier le sommet 1, puis on cherche dans la liste le prochain sommet qui soit non adjacent et on le colorie, soit le sommet 4.
La liste devient $3 - 5 - 2$.
3 et 2 sont colorés avec une nouvelle couleur d'usage soit le bleu.
Le sommet 5 est coloré avec la dernière couleur d'usage soit le jaune. Le graphe de la figure **??**.

2 Chapitre 2 : Chaines, chemins et cycles

Dans un graphe il est naturel de vouloir se déplacer de sommet en sommet en suivant les arêtes. Une telle marche est appelée une chaine ou un chemin. Un certain nombre de questions peuvent alors se poser : pour 2 sommets du graphe, existe-t-il un chemin pour aller de l'un à l'autre ? Quel est l'ensemble des sommets que l'on peut atteindre depuis un sommet donné ? Comment trouver le plus court chemin pour aller d'un sommet à un autre ?
Un chemin ou une chaine est une liste de sommets telle qu'il existe dans le graphe une arête entre chaque paire de sommets successifs.
La longueur du chemin ou de la chaine correspond au nombre d'arêtes parcourues : pour k sommets, la longueur est de k -1.

2.1 Graphe non orienté- Chaînes & cycles

Dans un graphe non orienté, Une « chaîne » dans un graphe est une succession d'arêtes qui permet de relier deux sommets du graphe.

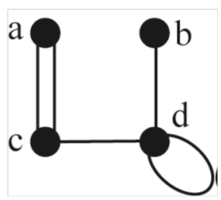


Figure 13 – Exemple d'un graphe non orienté à 4 sommets

Une chaîne qui ne passe qu'au plus une fois par chaque sommet est dite **élémentaire**.
La seule chaîne élémentaire reliant les sommets b et c dans l'exemple de la figure 13 est la chaîne constituée des arêtes (b,d) et (c,d).
Une chaîne qui ne passe qu'au plus une fois par chaque arête est dite **simple**.
Il n'existe que quatre chaînes simples reliant b et c dans le graphe ci-dessus. Elles sont constituées des séquences suivantes d'arêtes :

- b, d, c.
- b, d, d, c.
- b,d, c, a, c.
- b, d, d, c, a, c.

Un **cycle** est une chaîne dont les extrémités coïncident, càd, une chaîne dont l'extrémité initiale est égale à l'extrémité finale.
On parle de **cycle élémentaire** lorsque les sommets du graphe ne sont visités qu'au plus une fois, et on parle de cycle **simple** si aucune arête n'est utilisée deux fois, un exemple est illustré dans la figure 14.
Cycle élémentaire : a-b-c-d-e-f-a.
Cycle simple : a-b-c-d-e-f-b-d-f-a.

2.2 Graphe orienté – Chemins & circuits

Un **chemin** dans un graphe orienté est une succession d'arcs qui permet de se rendre d'un sommet à un autre.
Un **circuit** dans un graphe orienté est un chemin dont l'extrémité initiale coïncide avec l'extrémité finale.
Les chemins et les circuits peuvent être élémentaires ou simples, comme dans le cas des chaînes et des cycles. Ils sont élémentaires s'ils ne passent pas deux fois par un même sommet et ils sont simples s'ils ne passent pas deux fois par un même arc.

2.3 Algorithme pour tester l'absence de circuits

1. Marquer tout sommet sans successeur,
2. Marquer tout sommet dont tous les successeurs sont marqués.
3. Si on parvient à marquer tous les sommets, le graphe est sans circuits.

2.4 Autres types de chaine et chemin

1. Eulérien :
   Chemin eulérien est un chemin simple qui passe par tous les arcs.
   Chaîne eulérienne est une chaîne simple qui passe par toutes les arêtes.
2. Hamiltonien :
   Chemin hamiltonien est un chemin élémentaire qui passe par tous les sommets.
   Chaîne hamiltonienne est une chaîne élémentaire qui passe par tous les sommets.

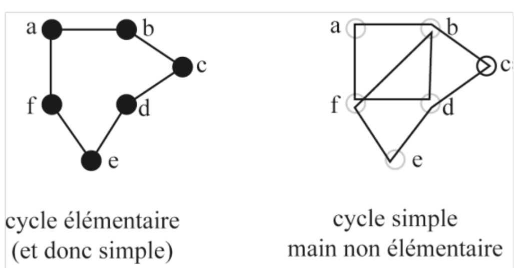


FIGURE 14 – Cycle élémentaire- cycle simple

Le graphe de la figure 15 est eulérien car il admet au moins un cycle eulérien : (1, 2, 3, 4, 2, 5, 6, 1, 5, 4, 1)
En voilà un autre : (1, 4, 2, 5, 1, 6, 5, 4, 3, 2, 1)

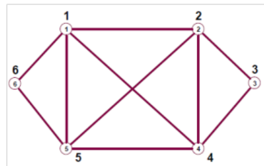


FIGURE 15 – Cycle élémentaire

2.5 La connexité

2.5.1 Graphe connexe

Un graphe est connexe s'il est possible à partir de n'importe quel sommet, de rejoindre tous les autres en suivant les arêtes. En d'autre terme, Un graphe G est connexe s'il existe au moins une chaîne entre une paire quelconque de sommets de G.
La relation de la formule 6 est une relation d'équivalence (réflexivité, symétrie, transitivité). Les classes d'équivalence induites sur X par cette relation forment une partition de X en $C_1, C_2, ..., C_p$.

$$x_i \text{ R } x_j \Leftrightarrow \begin{cases} \text{soit } x_i = x_j \\ \text{soit il existe une chaine joignant } x_i \text{ à } x_j \end{cases} \tag{6}$$

Le nombre p de classes d'équivalence distinctes est appelé nombre de connexité du graphe.
On peut alors donner une autre définition concernant la connexité d'un graphe. Un graphe est dit connexe si et seulement si son nombre de connexité est égal à 1.
Les sous-graphes $G_1, G_2, ..., G_p$ engendrés par les sous-ensembles $C_1, C_2, ..., C_p$ sont appelés les composantes connexes du graphe. Chaque composante connexe est un graphe connexe.

2.5.2 Graphe fortement connexe

Un graphe orienté est dit fortement connexe s'il existe un chemin joignant deux sommets quelconques. La relation de la formule 7 est une relation d'équivalence (réflexivité, symétrie, transitivité). Les classes d'équivalence induites sur X par cette relation forment une partition de X en $C_1, C_2, ..., C_p$.

$$x_i \text{ R } x_j \Leftrightarrow \begin{cases} \text{soit } x_i = x_j \\ \text{soit il existe à la fois un chemin joignant } x_i \text{ à } x_j \text{ et un chemin joignant } x_j \text{ à } x_i \end{cases} \tag{7}$$

Les sous-graphes $G_1, G_2, ..., G_p$ engendrés par les sous-ensembles $C_1, C_2, ..., C_p$ sont appelés les composantes fortement connexes du graphe. Chaque composante connexe est un graphe connexe.
Soit le graphe de la figure 16. Les composantes fortement connexes sont : $C_1 = \{1,\ 2,\ 3\}$ $C_2 = \{5,\ 6,\ 7,\ 8\}$ $C_3 = \{4\}$. Soit G un graphe oriente admettant p

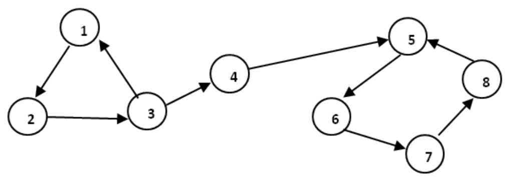


FIGURE 16 – Exemple de composantes fortement connexes

composantes fortement connexes : $C_1, C_2, ..., C_p$. On définit le graphe réduit de G notée Gr par : $G_r = (X_r, U_r)$ avec $X_r = C_1, C_2, ..., C_p$ et $(C_i, C_j) \in U_r$.
Il existe au moins un arc dans G ayant son extrémité initiale dans la composante fortement connexe C_i et son extrémité terminale dans la composante fortement connexe C_j. Le graphe de la figure 17 est le graphe réduit de celui de la figure

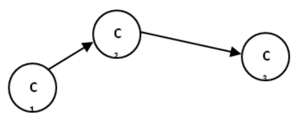


FIGURE 17 – Exemple de composantes fortement connexes

16, c'est le graphe dont les sommets sont les composantes fortement connexes (dites maximales) qui comportent le plus de sommets, et les arcs les liens entre celles-ci. Le graphe réduit ne contient pas de circuits, mais peut contenir des cycles.

2.5.3 Algorithme de recherche d'une composante fortement connexe

Algorithm 1 Algorithme Principal

Titre : RechercheCFC ;
Entrée : $G = (X; U)$ un graphe ;
Sortie : C=$\{C_1, ..., C_n\}$ un ensemble de composantes connexes ;
Variables intermédiaire : $\grave{X}$ un sous ensemble de sommets, i un entier, x un nœud ;
Début
$\grave{X} \leftarrow X$;
$i \leftarrow 1$;
while $(\grave{X} \neq \emptyset)$ **do**
  Choisir x dans $\grave{X} = X$;
  $C_i \leftarrow CFC(G, x)$;
  $\grave{X} \leftarrow \grave{X} - C_i$;
  $i \leftarrow i + 1$;
end while

L'algorithme fonctionne selon les étapes suivantes :

- Choisir un sommet a, au départ aucun sommet n'est examiné.
- Parcourir le graphe à partir de a et créer un ensemble de sommets parcourus $X1$.
- Même chose mais dans le sens indirect, créer un deuxième ensemble de sommets parcourus $X2$.
- L'intersection $X1 \cap X2$ donne les sommets qui à la fois peuvent atteindre a et sont accessible à partir de a.
- $X1 \cap X2 =$ la CFC qui contient a.

Algorithm 2 Fonction CFC

Titre : CFC ;
Entrée : $G = (X; U)$ un graphe ; a un sommet ;
Sortie : $\grave{X}$ un sous ensemble de sommets ;
Variables intermédiaire : $X1$ et $X2$ deux sous-ensemble de sommets, $examiner()$ une fonction, x un nœud.
Début
$X1 \leftarrow a$;
Pour tout $x \in X$ faire $examiner(x) \leftarrow faux;$;
while $\exists x \in X1/non\_examiner(x)$ **do**
  $examiner(x) \leftarrow vrai$;
  pour tout $u = (x; y) \in U/y \notin X1$ faire $X1 \leftarrow X1 \cup y$;
end while
$X2 \leftarrow a$;
Pour tout $x \in X1$ faire $examiner(x) \leftarrow faux$;
while $\exists x \in X2/non\_examiner(x) faire$ **do**
  $examiner(x) \leftarrow vrai$;
  pour tout $u = (y; x) \in U/y \notin X2$ faire $X2 \leftarrow X2 \in y$;
end while$\grave{X} \leftarrow X1 \cap X2$;
Fin

3 Chapitre 3 : Les arbres

Un arbre est un graphe connexe sans cycle. La figure 18 représente un exemple d'un arbre T. Un arbre $T = (S, U)$ comporte $|S| - 1$ arêtes.

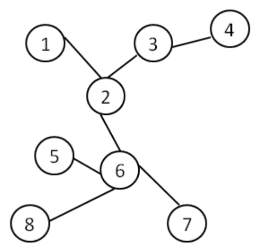


Figure 18 – Exemple d'un arbre T

Un arbre est un graphe simple.
T est une arborescence s'il existe une racine appelée R.
Un arbre couvrant est un graphe partiel qui soit aussi un arbre, voir l'exemple de la figure 19. T couvre tous les sommets de G.

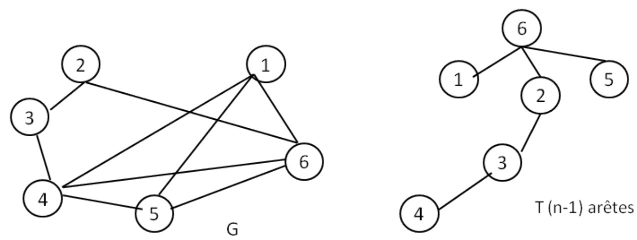


Figure 19 – Arbre couvrant

3.1 Problème de graphe couvrant de poids minimal : ACM

Cet algorithme consiste à trouver un arbre couvrant dont la somme des poids des arêtes soit minimale.

Algorithme de KRUSKAL

$G = (X, U)$ avec $|X| = N$ et $|U| = M$, $T = \emptyset$.

1. Classer les arêtes par ordre croissant (selon leurs poids) ;
2. Si $T + U_i$ est sans cycle alors ajouter U_i à T.
   $T \leftarrow T + U_i$.
   (U_i doit être incident à une des arrêtes de T).
   sinon aller à 3.
3. Si $|T| = N - 1$ STOP.
   sinon aller à 2.

Application de l'algorithme de KRUSKAL

1. Classification : Classer les arêtes selon leurs poids par ordre croissant : {AB, BC, AC, AD, BD}.
2. $AB = 1$; T={AB}, $|T| = 1$
   $BC = 1$; T={AB, BC}, $|T| = 2$
   $AD = 4$; T={AB, BC, AD}, $|T| = 3$; STOP.

Algorithme de PRIM

L'algorithme de PRIM est un autre type d'algorithme ACM. Il permet la construction incrémentale d'un arbre de poids minimum.
L'algorithme de Prim fait pousser un arbre couvrant minimal en ajoutant au sous-arbre T déjà construit une nouvelle branche parmi les arêtes de poids minimal joignant un sommet de T à un sommet qui n'est pas dans ce dernier. L'algorithme s'arrête lorsque tous les sommets du graphe sont dans T.
Soit X l'ensemble de sommets du graphe de départ G. On utilisera un ensemble d'arêtes qui sera en sortie l'arbre couvrant en question, et S un ensemble qui contiendra les sommets de T.
Soit le graphe représenté dans la figure 20, l'application de l'algorithme de Prim donne les différentes étape représenté dans la figure 21

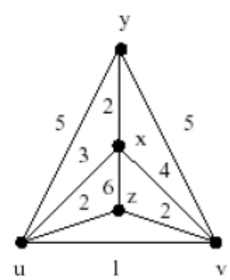


FIGURE 20 – Graphe G-Prim

Algorithm 3 PRIM

$T = \emptyset$
$S = S \cap x$
Tant que $S \neq X$ faire
trouver une arête e=y,s de poids minimal tel que $y \in X - Sets \in S$
$T = T \cap y, s$
$S = S \cap y$
FinTantque

La figure 21 montre les différentes étape pour construire l'arbre T.

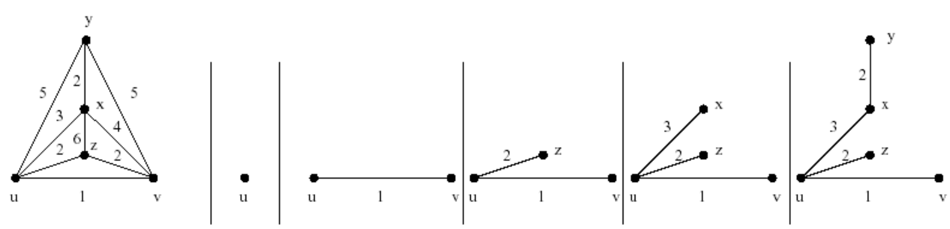


FIGURE 21 – Les différentes étape pour construire l'arbre T

4 Chapitre 4 : Problème de recherche du plus court chemin

Dans un graphe, il est naturel de se déplacer d'un point A vers un autre point B. Plusieurs chemins peuvent relier ces deux points. Le but de ce chapitre est d'en trouver le plus court chemin en terme de distance, de temps, de coût,... Dans ce chapitre, nous présentons deux algorithme de recherche du plus court chemin, à savoir : *Algorithme de Dijkstra* et celui de *Bellman Ford*.

4.1 Algorithme de Dijkstra

Algorithm 4 Dijkstra

G un graphe.
X={1,2,...,N} ensemble des sommet du graphe.
S ensemble de sommets définitivement traités.
$\bar{S}$ ensemble des sommets provisoirement traités.
P liste des prédécesseurs.
Π liste des distances.
Hypothèse : Le sommet du départ est le sommet 1.
a) Initialisation :
$S = 1$
$\bar{S} = X - S$={2,3,...,N}
$\Pi(1) = 0$

$$\Pi(i) = \begin{cases} l_{ij} & \text{si } \{i\} \in \Gamma^+(1) \\ \infty & \text{sinon.} \end{cases}$$

$P(i) = 1$
b) Sélectionner $j \in \bar{S}$ tel que $\Pi(j) = min\Pi(i)$ avec $i \in \bar{S}$ faire
$\bar{S} \leftarrow \bar{S} - j$
$S \leftarrow S + j$
Si $S = \emptyset$ alors FIN
sinon aller à c)
Pour tout $i \in \Gamma^+(j)$ et $i \in \bar{S}$ Faire
$\Pi(i) \leftarrow min(\Pi(i), \Pi(j) + l_{ij})$
Si $\Pi(i)$ est modifié alors $P(i) = j$
Retourner à b)

4.2 Application de l'algorithme de Dijkstra

Soit le graphe représenté dans la figure 22.
L'application de l'algorithme de Dijkstra se fait de la manière suivante :

1. Initialisation :
   $S = A$, $\bar{S}$={B,C,D,E,F}

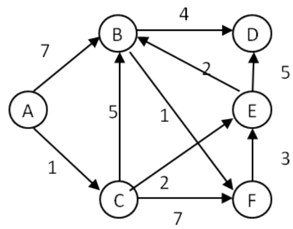


FIGURE 22 – Exemple du plus court chemin

$\Pi(A) = 0$, $\Pi(B) = l_{AB} = 7$, $\Pi(D) = \Pi(E) = \Pi(F) = \infty$, $\Pi(C) = l_{AC} = 1$
$P(B) = P(C) = A$. A chaque calcul, nous remplissons le tableau 1.

2. Sélectionner $j \in \bar{S}$ tel que $\Pi(i)$ soit minimale : $j = C$.
   S={A,C}, $\bar{S}$={B,D,E,F}
   $\Pi(B) = min(\Pi(B), \Pi(C) + l_{CB}) = min(7, 6) = 6$
   $\Pi(E) = min(\Pi(E), \Pi(C) + l_{CE}) = min(\infty, 1 + 2) = 3$
   $\Pi(F) = min(\Pi(F), \Pi(C) + l_{CF}) = min(\infty, 7 + 1) = 8$

Après une suite d'itération, nous obtenons les résultats représentés dans le tableau 1. A partir du tableau, le plus court chemin de A à D est : A-C-E-D, sa valeur étant de 8.

TABLE 1 – Tableau de Dijkstra

S	$\bar{S}$	$\Pi(A)$	$\Pi(B)$	$\Pi(C)$	$\Pi(D)$	$\Pi(E)$	$\Pi(F)$	$P(A)$	$P(B)$	$P(C)$	$P(D)$	$P(E)$	$P(F)$
A	B, C, D, E, F	0	7	1	∞	∞	∞	/	A	A	/	/	/
A, C	B, D, E, F	0	6	1	∞	3	8	/	C	A	/	C	C
A, C, E	B, D, F	0	5	1	8	3	8	/	E	A	E	C	C
A, C, E, B	D, F	0	5	1	8	3	6	/	E	A	E	C	B
A, C, E, B, F	D	0	5	1	8	3	6	/	E	A	E	C	B
A, C, E, B, F, D	$\emptyset$	0	5	1	8	3	6	/	E	A	E	C	B

4.3 Algorithme de Bellman Ford

4.4 Application de l'algorithme de Bellman Ford

L'application de l'algorithme de Bellman Ford sur le même graphe représenté dans la figure 22, se fait de la manière suivante (tout en remplissant le tableau 2) :
Initialisation :
$\Pi(A) = 0$ $\Pi(B) = \Pi(C) = \Pi(D) = \Pi(E) = \Pi(F) = \infty$.

1. Itération 1 :
   Sommet B : $\Gamma^{-}(B)$= {A,C,E}

Algorithm 5 Algorithme de Bellman Ford

G un graphe
$X = 1, 2, ..N$ ensemble des sommets du graphe
P liste des prédécesseurs.
Π liste des distances.
Hypothèse : Le sommet du départ est le sommet 1.
a) Initialisation :
$\Pi(1) = 0$
Pour $i = 2$ à N Faire
$\Pi(i) = \infty$
b) Calculer
Pour $i = 2$ à N Faire
$\Pi(i) \leftarrow min(\Pi(i), min(\Pi(j) + l_{ji}))$ avec $j \in \Gamma^{-}(i)$
Si $\Pi(i)$ est modifié alors $P(i) = j$
Retourner à b)
L'algorithme s'arrête lorsque le système se stabilise.

$\Pi(B) = min(\Pi(B), min(\Pi(A){+}l_{AB}, \Pi(C){+}l_{CB}, \Pi(E){+}l_{EB})) = min(\infty, min(7, \infty, \infty)) = 7$.
Sommet C : $\Gamma^{-}(C)$={A}
$\Pi(C) = min(\Pi(C), \Pi(A) + l_{AC} = min(\infty, 1) = 1$.
Sommet D : $\Gamma^{-}(D)$={B,E}
$\Pi(D) = min(\Pi(D), min(\Pi(B) + l_{BD}, \Pi(E) + l_{ED} = min(\infty, 7+4, \infty)) = 11$.
Sommet E : $\Gamma^{-}(E)$={F,C}
$\Pi(E) = min(\Pi(E), min(\Pi(C){+}l_{CE}, \Pi(F){+}l_{FE} = min(\infty, 1{+}2, \infty)) = 3$.
Sommet F : $\Gamma^{-}(F)$={B,C}
$\Pi(F) = min(\Pi(F), min(\Pi(B) + l_{BF}, \Pi(C) + l_{CF} = min(\infty, 8, 8)) = 8$.

2. Itération 2 :
$\Pi(B) = min(7, 7, 6, 5) = 5, \Pi(C) = min(1, 1) = 1, \Pi(D) = min(11, 9, 8), \Pi(E) = min(3, 3, 11) = 3, \Pi(F) = min(8, 6, 8) = 8$.
3. Itération 3 :
$\Pi(B) = min(5, 7, 6, 5) = 5, \Pi(C) = min(1, 1) = 1, \Pi(F) = 6$.

Table 2 – Tableau de Bellman Ford

	$\Pi(A)$	$\Pi(B)$	$\Pi(C)$	$\Pi(D)$	$\Pi(E)$	$\Pi(F)$	$P(A)$	$P(B)$	$P(C)$	$P(D)$	$P(E)$	$P(F)$
Initial	0	∞	∞	∞	∞	∞	/	/	/	/	/	/
itérat1	0	7	1	11	3	8	/	A	A	B	C	B
itérat2	0	5	1	8	3	6	/	E	A	E	C	B
itérat3	0	5	1	8	3	6	/	E	A	E	C	B

Le système s'est stabilisé, et à partir du tableau, le plus court chemin de A à D est : A-C-E-D, sa valeur étant de 8.

5 Chapitre 5 : Le problème de flot

5.1 Le réseau de transport

Un réseau de transport est un graphe $G = (X, U, C(U))$ dans lequel les conditions suivantes sont respectées :

- Il existe un sommet source $S/\Gamma^{-}(S) = \emptyset$.
- Il existe un sommet puits $S/\Gamma^{+}(S) = \emptyset$.
- G est orienté et connexe.

Chaque arc est muni d'une valeur $C(U)$ appelée la *capacité de l'arc*.

5.2 Le flot dans un réseau de transport

Un flot représente l'acheminement d'un flux de matière depuis la source S vers une destination T avec les conditions suivantes :

- Pour tout $u \in U, 0 \leq f(u) \leq C(U)$.
- Pour un sommet intermédiaire, la somme des flots entrants = la somme des flots sortant. $\forall x \neq S, T : \sum_{U \in \Gamma^{-}x} f(U) = \sum_{U \in \Gamma^{+}x} f(U)$.
- La somme des flots sortant de S = la somme des flots entrant à T. $\sum_{U \in \Gamma^{+}S} f(U) = \sum_{U \in \Gamma^{-}T} f(U)$.

Le flot compatible

Dans un réseau de transport, un flot est compatible, si pour tout arc u, le flux qui le traverse ne dépasse pas sa capacité.

Le flot complet

Un arc u se dit saturé si $f(u) = C(u)$. Un chemin est saturé si un de ses arc est saturé. La figure 23 représente la saturation d'un réseau de transport (les flèches pointillées représentent un chemin saturé).

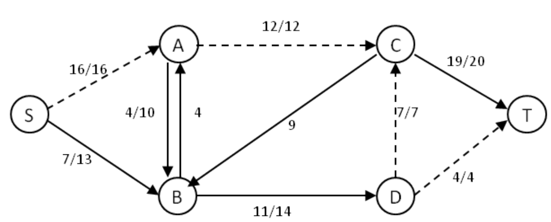


FIGURE 23 – Exemple d'un réseau de transport complet

- Le chemin $S - A - C - T$ est saturé par un flot de 12.
- Le chemin $S - A - B - D - T$ est saturé par 4.
- Le chemin $S - B - D - T$ ne peut être emprunté car il est déjà saturé.

– Le chemin $S - B - D - C - T$ est saturé par $min(13, 10, 7, 8) = 7$.
Nous obtenons un flot complet et sa valeur est de 23 (12+4+4+7).
Un flot est complet si pour tout chemin allant de la source au puits, il existe au moins un arc saturé.

5.3 Flot maximal

Le problème du flot maximal consiste à trouver le flot maximal possible de S à T.

Algorithme de Ford-Fulkerson

Initialement $f(U) = 0, \forall U$.

1. Tant qu'il existe un chemin P non saturé de S à T faire
   Saturer P
2. Tant qu'il existe une chaîne c augmentante S à T faire
   Augmenter c

L'augmentation de c se fait de la manière suivante :
- *Arc Direct :* augmenter si $f(U) < C(U)/U \in C^+$.
- *Arc Inverse :* augmenter si $f(U) > 0/U \in C^-$.
- augmenter par $\epsilon/\epsilon = min(C(U) - f(U)/U \in C^-, f(U)/U \in C^+)$.
- ajouter ϵ au flots des arcs directs, et l'enlever des flots des arcs inverses.

Soit le graphe de la figure 24, nous appliquons l'algorithme de Ford-Fulkerson sur ce graphe pour trouver le flot maximal.

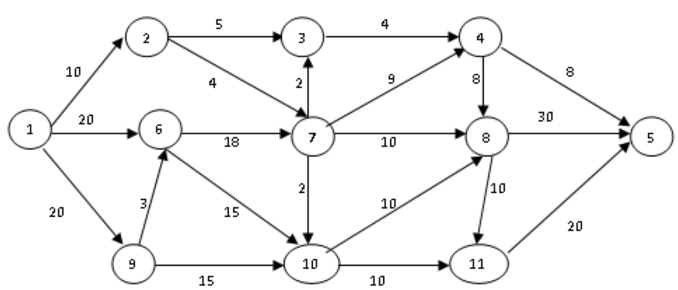


Figure 24 – Exemple de recherche de flot maximal

1) Saturer tous les chemins augmentants :
- 1-2-3-4-5 –> min(10,5,4,8)=4
- 1-6-7-8-5 –> min(20,18,10,30)=10
- 1-9-10-11-5 –> min(20,15,10,20)=10
- 1-6-10-8-5 –> min(10,15,10,20)=10
- 1-2-7-4-5 –> min(6,4,9,4)=4
- 1-9-6-7-4-8-5 –> min(10,3,8,5,8,10)=3

Nous avons obtenu un flot complet de valeur= 4+10+10+10+4+3=41

2) Trouver une chaine augmentant :

– 1-9-10-6-7-4-8-5 –> min(7,5,10,5,2,5,7)=2

Le nouveau flot est un flot maximal, sa valeur=41+2=43

6 Chapitre 6 : Planification de projet

La gestion d'un projet logiciel consiste à planifier sa progression, diriger, motiver et coordonner un groupe de professionnels.
Il est nécessaire d'avoir une visibilité du projet et de l'organiser afin de respecter le budget et le délais fixés au départ.
La méthode consiste à diviser le projet en plusieurs tâche et à assigner un rôle spécifique à chaque membre de l'équipe. Dans ce chapitre, nous présentons une des méthode d'ordonnancement très connue, il s'agit de la méthode Pert.
La méthode de Pert consiste à créer un diagramme indiquant les dépendances existantes entre les différentes tâches. A chaque tâche est attribuée une estimation du temps nécessaire à son achèvement ainsi que la liste des tâches qui doivent être terminées avant qu'elle ne puisse commencer.
Pour réaliser un diagramme de Pert, on se base sur les règles suivantes :

1. Une étape ne peut être exécutée avant que toutes les tâches précédente ne soient terminées.
2. Dans un diagramme de Pert, la date au plus-tôt est calculée par la formule 8, sachant que t_i est la tâche précédente et d_j sa durée.
$$t_i = max(t_i + d_j), avecj \in \omega^-(i) \tag{8}$$
3. Dans un diagramme de Pert, la date au plus-tard est calculée par la formule 9, sachant que tT_i est la tâche suivante et d_j sa durée.
$$T_i = min(T_i + d_j), avecj \in \omega^+(i) \tag{9}$$
4. Chaque étape est représenté comme le montre la figure 25.

Nom de l'étape	
Date au plus-tôt	Date au plus-tard

Figure 25 – Représentation d'une étape d'un diagramme de Pert

Soit l'exemple suivant présenté dans la table 3. La première colonne représente les tâches, la seconde, les tâches précédant celle-ci, et la dernière sa durée.

Pour dessiner le diagramme de Pert, nous commençons par définir les différents niveaux : $n_0 = A$; $n_1 = B, C$; $n_2 = D, E, F$; $n_3 = G, H, I$; $n_4 = J$; $n_5 = K$.
Le diagramme de Pert est illustré dans la figure 26

Table 3 – Tableau de tâches

Tâhce	Prédécesseur	Durée
A	/	1
B	A	2
C	A	3
D	B	1
E	C	1
F	C	3
G	D,E	2
H	F	1
I	F	2
J	G,H	2
K	j,I	2

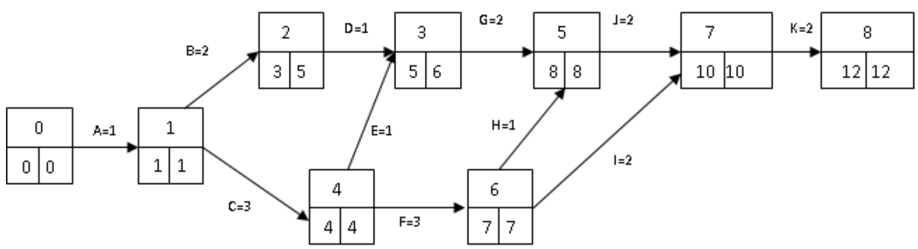


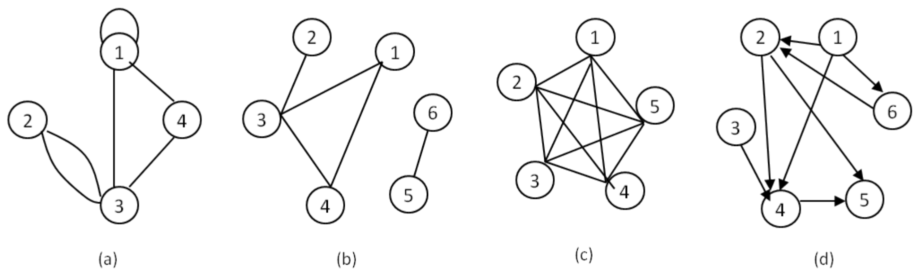
Figure 26 – Exemple de diagramme de Pert

Deuxième partie
Travaux dirigés

TD 1 : Notions de graphes

Exercice 1 :

Soit les graphes suivants :



1. Quels sont les graphes orientés et non orientés ?
2. Donner l'ensemble des sommets et des arcs ou arêtes.
3. Quel est l'ordre de chaque graphe.
4. Donner le degré de chaque sommet de chaque graphe.
   Considérant le graphe (d).
5. Donner la matrice d'adjacence et la table des successeurs et prédécesseurs.
   Donner la matrice d'incidence.

Exercice 2 :

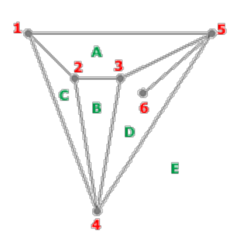
Soit la matrice d'incidence suivante, représenter graphiquement le graphe correspondant.

B=

1	1	0	0	0
-1	0	1	1	0
0	-1	-1	0	1
0	0	0	-1	-1

Exercice 3 :

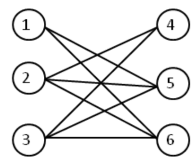
Soit le graphe planaire suivant : 1. Donner le degré de chaque face de ce graphe.



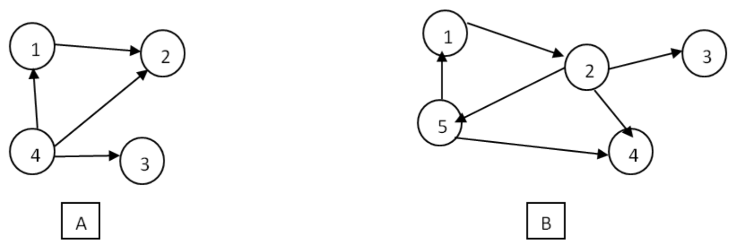
En déduire la formule d'euler.

Exercice 4 :

Donner le graphe planaire correspondant au graphe suivant :

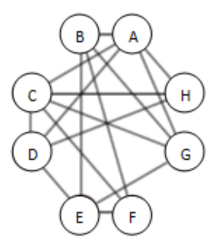


Exercice 5 : Vérifier l'existence ou l'absence de circuits dans les graphes suivants en utilisant l'algorithme de marquage.



Exercice 6 :

On veut transporter des produits chimiques par le rail. A, B, C, D, E, F, G et H désignent huit produits chimiques. Dans ce graphe, un lien signifie que les produits ne peuvent pas être entreposés dans le même wagon, car il y aurait risque d'explosion :



Question : Quel nombre minimum de wagons faut-il ? que contient chaque wagon ? Justifiez votre réponse par l'utilisation d'un algorithme.

TD 1 : Correction

Exercice 1 :

1) Les graphes orientés : (d)
Les graphes non orientés : (a), (b), (c)
2) Graphe (a) :
$X = 1, 2, 3, 4$
$U = (1, 1), (1, 3), (1, 4), (2, 3), (2, 3), (3, 4)$
Graphe (b) :
$X = 1, 2, 3, 4, 5, 6$
$U = (1, 3), (1, 4), (2, 3), (3, 4), (5, 6)$
Graphe (c) :
$X = 1, 2, 3, 4, 5$
$U = (1, 2), (1, 3), (1, 4), (1, 5), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (4, 5)$
Graphe (d) :
$X = 1, 2, 3, 4, 5, 6$
$U = (1, 2), (1, 6), (1, 4), (2, 4), (2, 5), (3, 4), (4, 5), (6, 2), (3, 4), (3, 5), (4, 5)$
3) L'ordre de chaque graphe :
Graphe (a) : 4, Graphe (b) : 6, Graphe (c) : 5, Graphe (d) : 6.
4) Le Degré de chaque sommet :
Graphe (a) : d(1)=4, d(2)= 2, d(3)=4, d(4)=2
Graphe (b) : d(1)=2, d(2)=1, d(3)=3, d(4)=2, d(5)=1, d(6)=1
Graphe (c) : d(i)=4
Graphe (d) : d(1)=3, d(2)= 4, d(3)=1, d(4)=4, d(5)=2, d(6)=2 **5)** Graphe d :
Matrice d'adjacence :

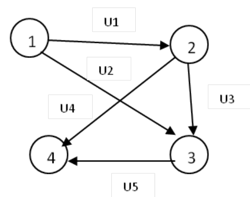
$$M = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Table des successeurs / prédécesseur :

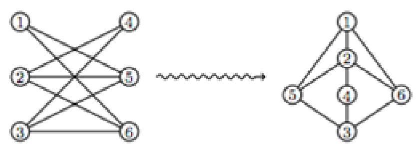
x_i	$\Gamma^+(x_i)$
1	2, 4, 6
2	4, 5
3	4
4	5
5	-
6	2

x_i	$\Gamma^-(x_i)$
1	-
2	6
3	-
4	2
5	2, 4
6	1

Exercice 2 :
Exercice 3 :
Le degré de chaque face :
d(A)=4, d(B)=3, d(C)=3, d(D)=5, d(E)=3.



Exercice 4 :
Le graphe planaire correspondant :



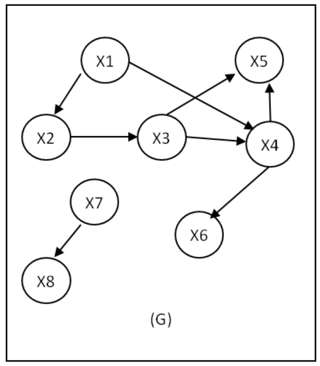
Exercice 5 :
Vérification de l'existence de circuit :
- Graphe A : marquer 3 et 2 puis 1 puis 4 (sans circuit).
- Graphe B : marquer 3 et 4, on peut pas 2 ni 5, il existe un circuit (1 ,2 ,5).

Exercice 6 :
Utilisation de Algorithme de k-coloration.
1) Trier les sommet par ordre décroissant des degrés :
- Liste : A,C,B,D,E,G,F,H
- Wagon 1={A,E}
- Nouvelle liste : C,B,D,G,F,H
- Wagon 2={C,B}
- Nouvelle liste : D,G,F,H
- Wagon 3={D,G,F}
- Nouvelle liste : H
- Wagon 4={H}

Il nous faut 4 wagons.

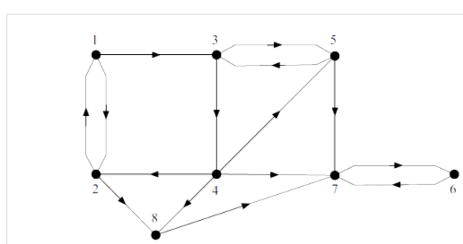
TD 2 : Cheminement et connexité

Exercice 1 :
Soit le graphe (G), Répondre au questions suivantes :



1. Donner une chaine élémentaire de longueur 4.
2. Donner une chaine démarrant de $x1$ et arrivant à $x3$, simple mais pas élémentaire de longueur 5.
3. Donner un cycle.
4. Supposant que xi représente des quartiers et xij des routes reliant ces derniers, donner un graphe partiel de G.
5. Donner un exemple de sous graphe de G.
6. Donner une chaine élémentaire qui n'est pas simple.
7. Ce graphe est-il fortement connexe ? pourquoi ?
8. Ajouter deux arcs de sorte que le graphe G admette un chemin hamiltonien. Quels sont les arcs ? Quel est ce chemin ?
9. Ajouter un arc pour que le graphe admette une chaîne eulérienne . Quel est cet arc ? Quelle est cette chaine ?

Exercice 2 :
Soit le graphe suivant :



Trouver les composantes fortement connexes de ce graphe et donner son graphe réduit.

TD 2 : Correction

Exercice 1 :

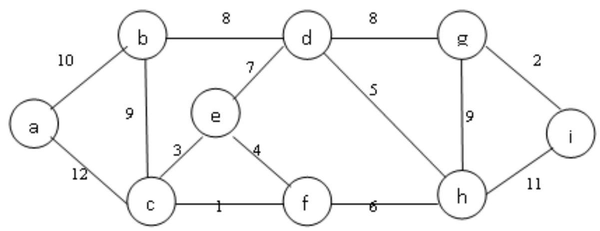
1. x1-x2-x3-x4-x6
2. x1-x2-x3-x4-x5-x3
3. x1-x4-x3-x2-x1
4.
5.
6. inexistante
7. non, car il n'est pas connexe.
8. (x6,x7) et (x8,x5).
9. x6,x7 –> La chaîne est : x3-x5-x4-x1-x2-x3-x4-x6-x7-x8

Exercice 2 :
Application de l'algorithme des composantes fortement connexes :
$\grave{X}$={1,2,3,4,5,6,7,8}
$i \leftarrow 1$

- Choisir 1 :
  $C_1 = CFC(G, 1)$
  $X_1 \leftarrow 1$
  Pour tout $U = (1, y)$ $U = (1, 2)$
  $X_1 \leftarrow X_1 \cup \{2\}$.
  $U = (1, 3)$
  $X_1 \leftarrow X_1 \cup \{3\}$.
- Choisir 2 :
  Pour tout $U = (2, y)$: $U = (2, 1)$, $U = (2, 8)$
  $X_1 \leftarrow X_1 \cup \{8\}$.
- on continu de la même manière .... X_1={1,2,3,4,5,6,7,8}
- Choisir 1 :
  $X_2 \leftarrow 1$
  Pour tout $U = (y, 1)$ $U = (2, 1)$
  $X_2 \leftarrow X_2 \cup \{2\}$.
- Choisir 2 :
  Pour tout $U = (y, 2)$ $U = (1, 2)$, $U = (4, 2)$.
  $X_2 \leftarrow X_2 \cup \{4\}$.
- on continu de la même manière .... X_2={1,2,3,4,5}
  $\grave{X} = X_1 \cap X_2$={1,2,3,4,5}.
  C_1={1,2,3,4,5}.
- Les composantes fortement connexes sont :
  C_1={1,2,3,4,5}.
  C_2={6,7}.
  C_3={8}.

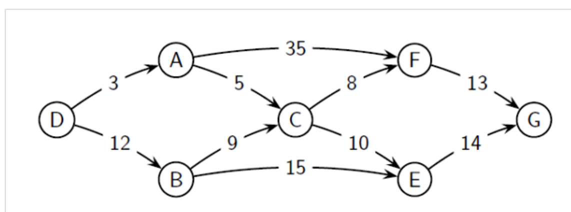
TD 3 : Arborescence et plus court chemin

Exercice 1 :
Un projet d'installation de conduits d'eau alimentant les localités : a,b,c,d,e,f,g,h,i est représenté par le graphe G=(X,U) suivant. Les valeurs portées sur les arêtes représentent la longueur des conduits utilisés pour l'installation.



Donner un plan d'installation minimisant la longueur totale des conduits d'eau utilisés.

Exercice 2 :
Déterminer le plus court chemin dans le graphe suivant en utilisant de Dijkstra, de la source à la destination.



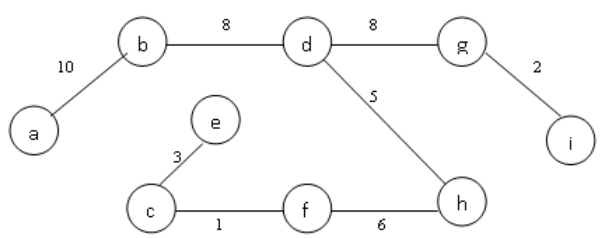
Exercice 3 :
Soit un graphe orienté, à 5 sommets, défini par la liste des arcs pondérés comme suit :
(A ;B) = 10, (A ;E) = 5 , (B ;E) = 2, (E ;B) = 3, (B ;C) = 1, (C ;D) = 4 , (E ;D) = 2, (E ;C) = 9, (D ;C) = 6 .

1. En appliquant l'algorithme de Bellman Ford, déterminer le plus court chemin à partir du sommet A vers les autres sommets.
2. Quel est le plus court chemin du sommet A vers le sommet C et quelle est sa valeur ?

TD 3 : Correction

Exercice 1 :
Application de l'algorithme de KRUSKAL :

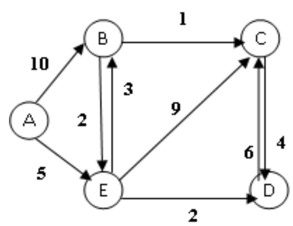
1. Classification
   - cf=1
   - gi=2
   - ce=3
   - ef=4
   - dh=5
   - fh=6
   - ed=7
   - bd=8
   - dg=8
   - gh=9
   - bc=9
   - ab=10
   - hi=11
   - ac=12
2. Application
   - T={cf}, $|T| = 1$;
   - T={cf,ce}, $|T| = 2$;
   - T={cf,ce,fh}, $|T| = 3$;
   - T={cf,ce,fh,dh}, $|T| = 4$;
   - T={cf,ce,fh,dh,bd}, $|T| = 5$;
   - T={cf,ce,fh,dh,bd,dg}, $|T| = 6$;
   - T={cf,ce,fh,dh,bd,dg,gi}, $|T| = 7$;
   - T={cf,ce,fh,dh,bd,dg,gi,ab}, $|T| = 8$;



Exercice 2 :
Application de l'algorithme de Dijkstra :
Le plus court chemin de D à G est : D-A-C-F-G, sa valeur est 29.

S	$\bar{S}$	$\Pi(A)$	$\Pi(B)$	$\Pi(C)$	$\Pi(D)$	$\Pi(E)$	$\Pi(F)$	$\Pi(G)$	$P(A)$	$P(B)$	$P(C)$	$P(D)$	$P(E)$	$P(F)$	$P(G)$
D	A, B, C, E, F, G	3	12	∞	0	∞	∞	∞	D	D	/	/	/	/	/
D, A	B, C, E, F, G	3	12	8	0	∞	38	∞	D	D	A	/	/	A	/
D, A, C	B, E, F, G	3	12	8	0	18	16	∞	D	D	A	/	C	C	/
D, A, C, B	E, F, G	3	12	8	0	18	16	∞	D	D	A	/	C	C	/
A, B, C, D, G	E, F	3	12	8	0	18	16	29	D	D	A	/	C	C	E

Exercice 3 :



Application de l'algorithme de Bellman Ford :

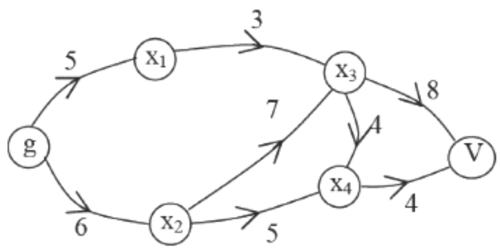
	$\Pi(A)$	$\Pi(B)$	$\Pi(C)$	$\Pi(D)$	$\Pi(E)$	$P(A)$	$P(B)$	$P(C)$	$P(D)$	$P(E)$
Initial	0	∞	∞	∞	∞	∞	/	/	/	/
itérat1	0	10	11	15	5	/	A	B	C	A
itérat2	0	8	9	7	5	/	E	B	E	A
itérat3	0	8	9	7	5	/	E	B	E	A

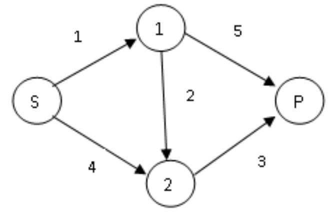
Le plus court chemin est : A-E-B-C, sa valeur est 9.

TD 4 : Les flots

Exercice 1 :

Trouver le flot maximum dans les graphes suivants :





Exercice 2 :

Avant d'établir un projet de construction d'autoroute, on désire étudier la capacité du réseau routier, représenté par la matrice R, reliant la ville $V1$ à la ville $V2$. Soient les propriétés suivantes :

R =

/	A	B	C	D	E	F	G	V2
V1	5	10				8		
A			7	10				
B			8	2		1		
C					7			
D					4		2	6
E								10
F				2			4	
G								6

- Les lignes représentent les sommets initiaux.
- Les colonnes représentent les sommets finaux.
- Un élément $r_{ij} \in R$ représente le nombre maximum de véhicules qui peuvent passer par une route (i, j) par heure. Ces évaluations sont données en centaine de véhicules par heure.

1. Dessiner le graphe correspondant à la matrice R.
2. Comment appelle-t-on r_{ij} ? $V1$? $V2$?
3. Quel est le débit horaire maximal de véhicules susceptibles de s'écouler entre les villes $V1$ et $V2$, et quel est l'algorithme utilisé ?

TD 4 : Correction

Exercice 1 :

Pour le graphe (1) :

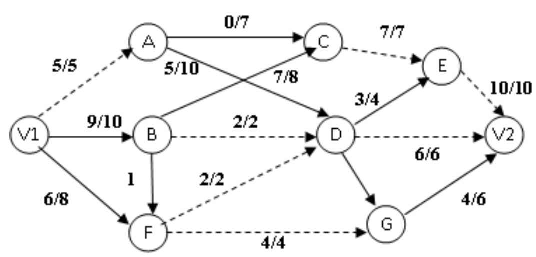
1. Saturer tous les chemins
   - g,x1,x3,v –> 3.
   - g,x2,x4,v –> 4.
   - g,x2,x3,v –> 2.
   - Flot complet de valeur =9.
2. Trouver une chaîne augmentante : il n'existe pas de chaîne augmentante.
   Flot maximal =9.

Pour le graphe (2) :

1. Saturer tous les chemins
   - s,1,2,p –> 1.
   - s,2,p –> 2.
   - Flot complet de valeur =3.
2. Trouver une chaîne augmentante :
   s,2,1,p –>min(2,1,5)=1.
   Le flot maximal=4.

Exercice 2 :

1)



2) r_{ij} est la capacité de l'arc, V1 représente la source, V2 est le puits.

3)

1. Saturer tous les chemins
   - V1-A-C-E-V2 –> 5.
   - V1-B-D-V2 –> 2.
   - V1-F-G-V2 –> 4.
   - V1-B-C-E-V2 –> 2
   - V1-F-D-V2 –> 2
   - Flot complet de valeur =15.
2. Trouver une chaîne augmentante :
   - V1-B-C-A-D-V2 –> min(6,6,5,10,2)=2.
   - V1-B-C-A-D-E-V2 –> min(4,4,3,8,4,3)=3.
   - La valeur du flot maximum est de 20, soit un débit de 2000 voitures par heure.

TD 5 : Ordonnancement

Exercice 1 :

Dessiner le graphe PERT correspondant à la table de tâches ci-dessous :

TACHES	PREDECESSEURS
A	-
B	A
C	A
D	B, C
E	B, C
F	D, E

TACHES	PREDECESSEURS
A	-
B	A
C	B
D	C, G
E	D, H, K, M
F	A
G	F
H	F
I	F
J	I, L
K	A
L	A
M	B

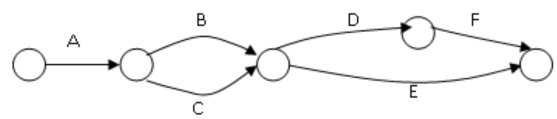
Exercice 2 :

Donner le diagramme de Pert correspondant avec les dates au plus tôt, dates au plus tard.

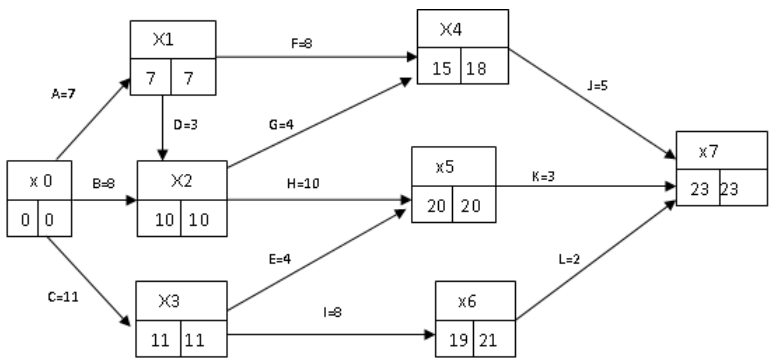
Tâhce	Prédécesseur	Durée
A	/	7
B	/	8
C	/	11
D	A	3
E	C	4
F	A	8
G	B,D	4
H	B,D	10
I	C	8
J	F,G	5
K	H,E	3
L	I	2

TD 5 : Correction

Exercice 1 :



Exercice 2 :



Références

[1] MULLER D. *Introduction à la théorie des graphes, CAHIERS DE LA CRM*, (2012)

[2] COGIS O., ROBERT C. *Théorie des graphes, Vuibert*, (2003)

[3] DROESBEKE F., HALLIN M., LEFEVRE C. *Les graphes par l'exemple, Ellipse*, (1987)

[4] GONDRAN M., MINOUX M. *Graphes et algorithmes, 4e édition, Lavoisier*, (2009)

[5] BELHARRAT N. *Théorie des graphe, Recherche opérationnelle, Pages bleues*