



مدرسة علوم المعلومات
ECOLE DES SCIENCES
DE L'INFORMATION

TSCHOOL.MA

ID 0132.002

Théorie des Graphes

Dr. CHERIF Walid

w.cherif@insea.ac.ma

Année universitaire : 2017 - 2018

Ce polycopié couvre les concepts centraux de la théorie des graphes. Il est conçu pour accompagner l'étudiant au fur et à mesure de l'avancement du cours magistral. Il est donc recommandé de prendre le temps de refaire les exercices types qui y sont proposés.

Le contenu peut également intéresser les professionnels, les chercheurs scientifiques, ou toute personne ayant une formation technique ou en ingénierie pour son auto-formation. Comme les applications de la théorie des graphes sont de plus en plus vastes, allant de la simple recherche du chemin optimal en logistique à des applications plus complexes telles que les propositions d'amis sur les réseaux sociaux, les personnes intéressées par ce cours sont très nombreuses.

La familiarité avec quelques éléments de base de mathématiques et d'informatique est exigée pour la bonne maîtrise du cours. Les conditions préalables à l'utilisation pratique ce polycopié, ainsi que ses interactions avec d'autres domaines mathématiques, sont ainsi notifiées par des rectangles verts en bas des pages :



Un ensemble de lacunes et d'erreurs fréquentes, signalées lors d'anciens examens et évaluations, sont signalées par des triangles rouges avec le symbole :



Dans chaque chapitre, des exercices d'application sont proposés pour évaluer la maîtrise des théorèmes et des algorithmes étudiés. Et à la fin du polycopié, une série d'exercices et de problèmes, ainsi que des extraits d'anciens examens et des passages de certains de mes travaux de recherche personnelle sont proposés. Le niveau de difficulté de cette série varie suivant le nombre d'étoiles sur la droite des énoncés:

- Une étoile : ★ *exercices simples, applications directes des théorèmes et des techniques du cours*
- Deux étoiles : ★★ *exercices moins faciles combinant application des théorèmes et un peu de réflexion*
- Trois étoiles : ★★★ *exercices difficiles nécessitant plus de réflexion ou une interaction avec des techniques d'autres disciplines*
- Quatre étoiles : ★★★★ *problèmes plus complexes, proposés pour une recherche individuelle ou en groupe.*

Vous êtes libres de réutiliser, après demande, le contenu de ce polycopié pour toute fin pédagogique.

Table des matières

Préface.....	2
Table des matières	3
Introduction	5
I. Généralités	7
I.1. Types de graphes	8
I.2. Décomposition d'un graphe	9
I.2.1. Graphe partiel	9
I.2.2. Sous-graphe	9
I.2.3. Chaîne et chemin	10
I.3. Mesures dans un graphe.....	11
I.3.1. Degré d'un sommet	11
I.3.1.a. Graphe non orienté	11
I.3.1.b. Graphe orienté	11
I.3.2. Distances	12
I.3.2.a. Distance entre deux sommets	12
I.3.2.a. Ecartement d'un sommet.....	13
I.3.2.a. Centre et rayon d'un graphe	14
I.3.2.a. Diamètre d'un graphe	14
I.4. Relations - graphe	15
I.4.1. Relations binaires	17
I.4.2. Matrice d'adjacence.....	18
I.4.3. Analogie avec les applications	19
I.4.4. Opérations sur les relations	20
I.4.4.a. Composition de relations.....	20
I.4.4.b. Inverse d'une relation	21
I.4.5. Relations dans un ensemble.....	21
I.4.5.a. Propriétés d'une relation.....	21
I.4.5.b. Relation d'équivalence	22
I.4.5.c. Relation d'ordre.....	23
I.5. Représentation matricielle	24
I.5.1. Définitions	24
I.5.1.a. Graphe non orienté	24

I.5.1.b.	Graphe orienté	24
I.5.2.	Matrice d'adjacence.....	25
I.5.2.a.	Graphe non orienté	25
I.5.2.b.	Graphe orienté	25
I.5.3.	Matrice d'incidence	27
I.5.3.a.	Graphe non orienté	27
I.5.3.b.	Graphe orienté	27
I.5.4.	Matrice laplacienne	28
I.6.	<i>Graphes particuliers</i>	29
I.6.1.	Graphe simple.....	29
I.6.2.	Graphe connexe	29
I.6.3.	Graphe complet	30
I.6.4.	Graphe biparti.....	30
I.6.5.	Graphe planaire	31
I.6.6.	Arbre.....	32
I.6.6.a.	Arbre non orienté.....	32
I.6.6.b.	Arborescence	36
I.6.7.	Graphe eulérien	37
I.6.8.	Graphe hamiltonien	38
I.7.	<i>Stabilité et absorption</i>	39
I.7.1.	Stabilité.....	39
I.7.2.	Absorption	39
I.8.	<i>Pondération d'un graphe</i>	40
I.8.1.	Graphe probabiliste	40
II.	Principaux algorithmes	42
II.1.	<i>Algorithme de coloration</i>	43
II.2.	<i>Algorithme des CFC</i>	45
II.3.	<i>Algorithme de Kruskal</i>	46
II.4.	<i>Algorithme de Prim</i>	48
II.5.	<i>Algorithme de Bellman</i>	50
II.6.	<i>Algorithme de Dijkstra</i>	53
II.7.	<i>Algorithme général</i>	55
II.8.	<i>Algorithme de Ford</i>	57

Introduction

Un graphe tire son nom du fait qu'on peut le représenter par un dessin, raison pour laquelle il est nécessaire de se munir d'un crayon et d'un brouillon pour comprendre la majorité des concepts de ce cours.

La théorie des graphes est la discipline mathématique et informatique qui étudie les graphes. Elle est très facile à lire et à interpréter mais sa conception demeure un point judicieux et complexe. Elle illustre le fait que les mathématiques ne sont pas toujours synonymes de modèles statistiques ou de calculs numériques, mais elles peuvent aussi cibler des représentations symboliques.

L'origine de la théorie des graphes remonte à 1735, lorsque le mathématicien suisse Leonhard Euler présenta à l'Académie de Saint-Petersbourg un article sur le problème des sept ponts de Königsberg (Euler L., 1741). Le problème consistait à trouver une promenade à partir d'un point donné qui fasse revenir à ce point en passant une fois et une seule par chacun des sept ponts de la ville de Königsberg ¹:

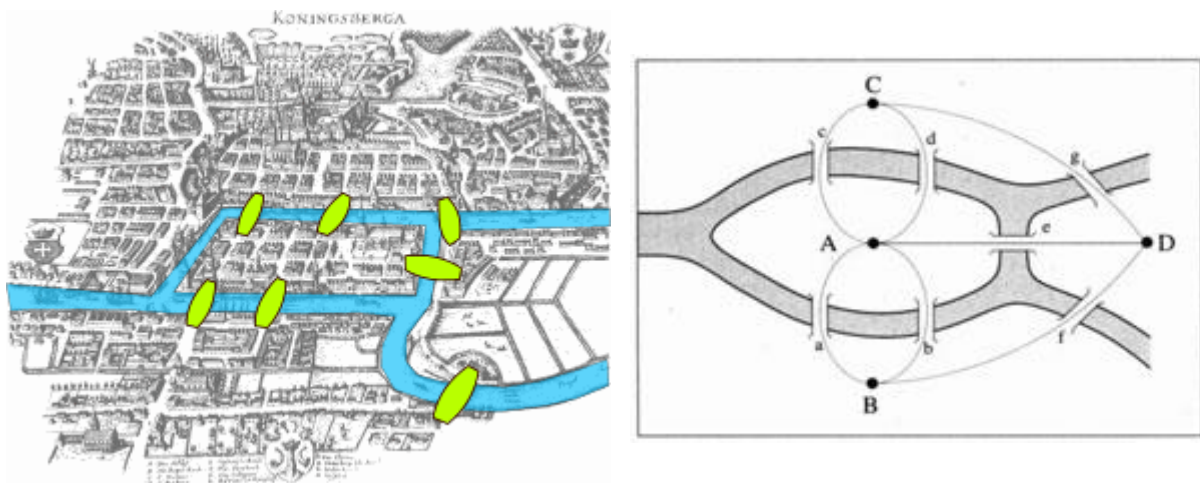


Fig. 1: modélisation des ponts de Königsberg par un graphe

En 1852, en étudiant si la carte des régions d'Angleterre pouvait être coloriée avec quatre couleurs de façon à ce que les régions limitrophes (ayant une frontière commune) aient des couleurs distinctes, le mathématicien sud-africain Francis Guthrie énonça le problème des quatre couleurs, qui fût démontré en 1976 par Kenneth Appel et Wolfgang Haken (Appel K., et Haken W., 1976).

L'étude de ce problème entraîna de nombreux développements en théorie des graphes.

¹ <https://goo.gl/maps/AgchNePKMK92>

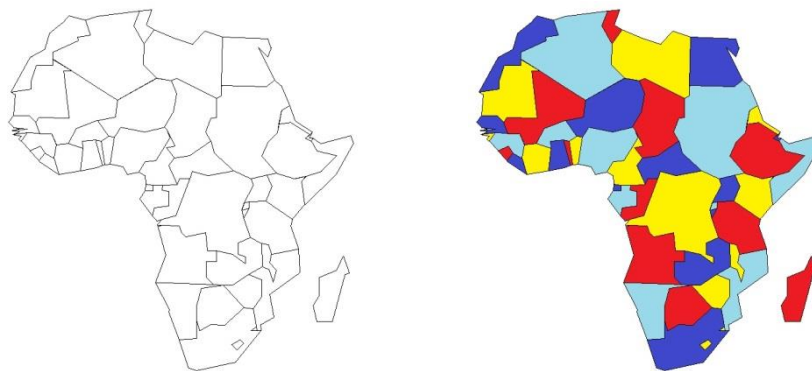


Figure 2: exemple du problème de coloration appliqué au continent africain

Ramsey poursuivit ces recherches sur la coloration, et en exploitant les résultats du mathématicien hongrois Pál Turán, il développa la théorie des graphes extrémaux. Ces derniers cherchent le maximum d'une quantité donnée sous certaines contraintes.

Des problèmes d'optimisation poussèrent les mathématiciens à la fin du XIX^{ème} siècle à s'intéresser à un nouveau type de graphes : les graphes couvrants qui ont la particularité de contenir tous les sommets mais seulement une partie des arcs. Des exercices sur ce type de graphes sont proposés dans ce polycopié.

Une autre application classique de la théorie des graphes est connue chez les physiciens: les lois de Kirchhoff qui expriment la conservation de l'énergie et de la charge dans un circuit électrique. On doit cette découverte au physicien allemand Gustav Kirchhoff qui a étendu le travail de Georg Ohm en 1845, pour ainsi représenter un circuit électrique sous la forme d'un graphe, dans lequel les sommets sont les nœuds du circuit, et les arcs sont les connexions physiques entre ces nœuds.

Mais c'est bien au XXI^{ème} siècle que la théorie des graphes a connu son véritable essor grâce notamment à l'évolution du secteur des réseaux avant de s'introduire dans notre vie quotidienne sous ses différentes formes (parfois sans que l'on ne s'en aperçoive), bénéficiant des avancées technologiques et de l'énorme puissance de calcul des machines actuelles:

- réseaux de transport de données : téléphonie, wifi, ...
- réseaux d'informations : bases de données, web, réseaux sociaux, ...
- réseaux logistiques : transport routier, transport de marchandise, ...
- le jeu de dessin qui consiste à tracer un carré avec ses diagonales sans lever la main et sans repasser deux fois sur le même trait,
- la fabrication d'un ballon de football (en pentagones noirs et blancs),
- la recherche du chemin le plus rapide pour se rendre d'une ville à une autre, ...

Cette théorie est devenue fondamentale en informatique car elle fournit de nombreux algorithmes pour résoudre des problèmes complexes représentés par des graphes de très grande taille.

I. Généralités

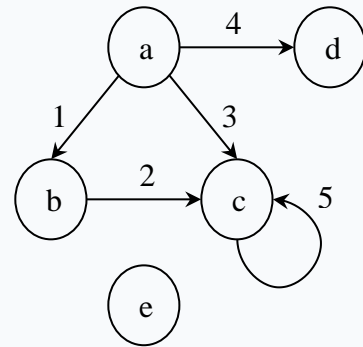
Un graphe est constitué d'un ensemble de sommets (noté X), et d'un ensemble d'arcs qui les connectent (noté U).

Un arc est un couple de sommets, il est donc un élément du produit cartésien $X \times X$.

Le graphe (noté par exemple G) est représenté par le couple (X, U) .

Exemple 01 : Composantes d'un graphe

- Ensemble des sommets : $X = \{a, b, c, d, e\}$
 - Les arcs :
 - $1 = (a, b)$
 - $2 = (b, c)$
 - $3 = (a, c)$
 - $4 = (a, d)$
 - $5 = (c, c)$
- L'ensemble des arcs est: $U = \{1, 2, 3, 4\}$



Le nombre de sommets est appelé : ordre du graphe, il est noté : $|G|$ ou simplement $n = \text{card}(X)$.

Dans l'exemple précédent : $|G| = 5$

Le nombre d'arcs est appelé : taille du graphe : $m = \text{card}(U)$.

Cette représentation qui associe les sommets à des points de l'espace et les arcs à des flèches allant d'un point à un autre, est appelée représentation sagittale du graphe.

Dans un arc (x, y) : x est l'extrémité initiale et y l'extrémité finale.

Dans l'arc : $3 = (a, c)$, a est l'extrémité initiale et c l'extrémité finale.

On dit aussi que c est un successeur de a ou que a est un prédécesseur de c .

L'ensemble des successeurs d'un point y est l'ensemble des points x_i pour lesquels les arcs (x_i, y) existent.

L'ensemble des successeurs de a est : $\{b, c, d\}$

De même, l'ensemble des prédécesseurs de c est : $\{a, b, c\}$

Un arc de la forme (x, x) est une boucle.

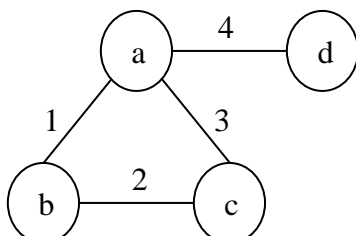
(c, c) est une boucle.

I.1. Types de graphes

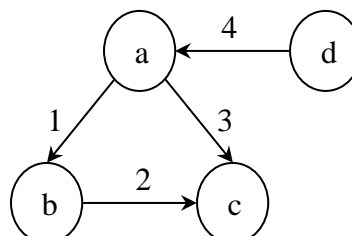
Certains problèmes font intervenir l'orientation, d'autres non. Nous distinguons alors deux types de graphes :



- Les graphes non orientés GNO



- Les graphes orientés GO



Dans un graphe non orienté, il nous est possible de passer d'un sommet à un autre tant qu'il y a un arc liant les deux sommets. Par contre, pour un graphe orienté, il faut de plus que la direction de l'arc soit du sommet de départ vers son successeur.

Un graphe non orienté peut ainsi être vu comme étant un cas particulier du graphe orienté, dans lequel les arcs sont doubles (ces arcs seront appelés arêtes):



Dans ce qui suit, nous allons définir un ensemble de notions fondamentales pour la suite du cours, ces définitions dépendent du type de graphe (orienté ou non orienté) raison pour laquelle nous les étudierons pour chacun des deux cas. Un graphe orienté sera noté parfois G^o pour le distinguer du graphe non orienté G .



Les algorithmes étudiés ainsi que certaines propriétés sont spécifiques à des types particuliers de graphes, il faut distinguer dès maintenant les différents types de graphes afin de déterminer l'algorithme adéquat à y appliquer.

Application 01 : Premiers graphes

- 01.1. Tracer un graphe non orienté G d'ordre 4 dont tous les sommets sont 2 à 2 adjacents (liés par une arête).
- 01.2. Tracer un graphe orienté G^o d'ordre 4 dont les sommets ont tous le même nombre de prédécesseurs et le même nombre de successeurs = 2.

I.2. Décomposition d'un graphe

I.2.1. Graphe partiel

Notons $G = (X, U)$ un graphe.

Le graphe $G_1 = (X, U_1)$ tel que $U_1 \subset U$ est appelé graphe partiel de G .

G_1 est obtenu en enlevant un ou plusieurs arcs/arêtes du graphe G .

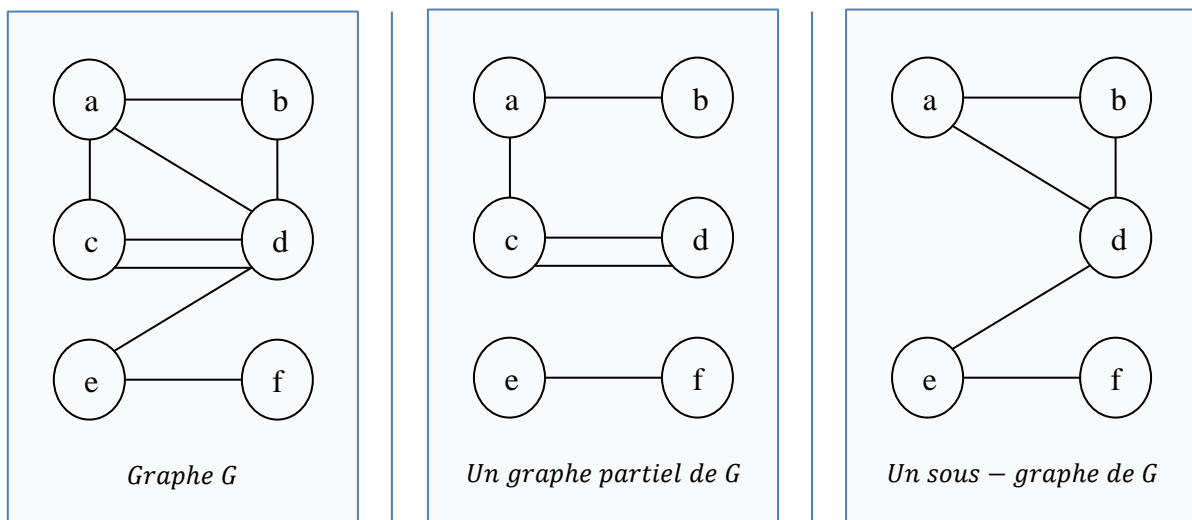
I.2.2. Sous-graphe

Notons X_2 un sous-ensemble de sommets inclus dans X , et U_2 l'ensemble des arcs/arêtes ayant leurs deux extrémités dans X_2 .

Le graphe $G_2 = (X_2, U_2)$ est appelé sous-graphe de G induit par X_1 .

G_2 est obtenu en enlevant un ou plusieurs sommets du graphe G , ainsi que tous les arcs/arêtes incidents à ces sommets.

Exemple 02 : Graphe partiel et sous-graphe



Un graphe partiel d'un sous-graphe est un sous-graphe partiel de G .

Application 02 : Suppression de sommets d'un graphe

- 02.1. Reprenons le graphe de l'*Application 01 : Premiers graphes*, question 1 :
Proposer un sous-graphe en supprimant un seul sommet.
- 02.2. Combien d'arêtes ont été supprimées ?
- 02.3. Combien d'arêtes auraient été supprimées si nous avions supprimé 2 sommets ?

I.2.3. Chaîne et chemin

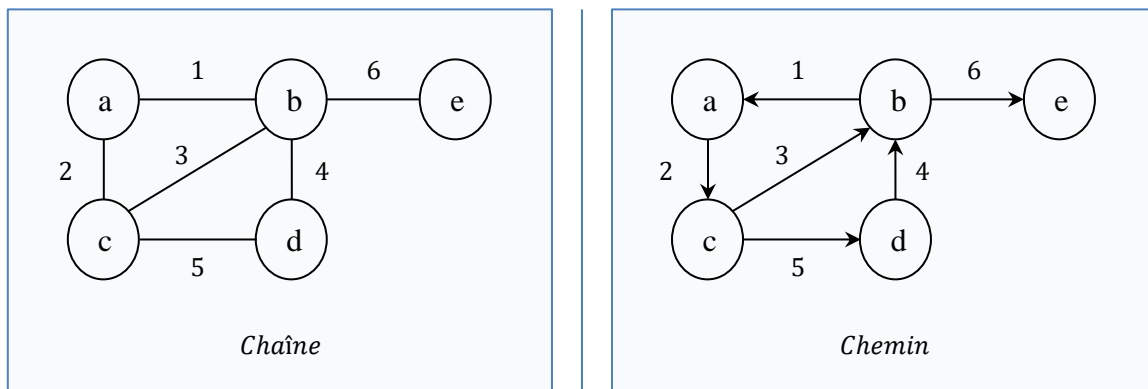
Dans un graphe non orienté G , une chaîne est une suite ayant pour éléments alternativement des sommets et des arêtes, commençant et se terminant par un sommet.

La chaîne relie un premier sommet : a à un dernier sommet : b en passant par des arêtes et des sommets intermédiaires. On la note : $\mu(a, b)$.

Le nombre d'arêtes parcourues est appelé longueur de la chaîne.

Lorsque le graphe G est orienté, la chaîne est appelée chemin :

Exemple 03 : Chaîne et chemin



La chaîne est décrite par énumération de ses éléments :

Dans la chaîne de l'*Exemple 03* :

$$\mu_1(a, e) = \{a, 1, b, 6, e\}$$

$$\mu_2(a, e) = \{a, 2, c, 3, b, 6, e\}$$
 sont deux chaînes possibles liant les sommets a et e .

Dans les deux chaînes, chaque sommet apparaît au plus une fois.

On les appelle : des chaînes élémentaires.

Ce n'est pas le cas de la chaîne : $\mu_3(a, e) = \{a, 1, b, 3, c, 5, d, 4, b, 6, e\}$ où le sommet b est parcouru 2 fois.

Les chaînes précédentes parcourent au plus une fois la même arête.

On les appelle : des chaînes simples.

Ce n'est pas le cas de la chaîne : $\mu_4(a, e) = \{a, 1, b, 3, c, 2, a, 1, b, 6, e\}$ où l'arête 1 est parcourue 2 fois.

Dans un chemin, tous les arcs doivent être parcourus dans le bon sens.

Dans le chemin de l'*Exemple 03* :

$\mu_1(a, e) = \{a, 1, b, 6, e\}$ n'est pas un chemin mais $\mu_5(a, e) = \{a, 2, c, 5, d, 4, b, 6, e\}$ en est un. Une chaîne simple qui boucle (les sommets de départ et d'arrivée sont identiques), est appelée un cycle.

Lorsque le graphe G est orienté, le cycle est appelé : circuit.

Dans l'*Exemple 03* :

Dans la chaîne : $\mu_1(a, a) = \{a, 1, b, 3, c, 2, a\}$ est un cycle.

Dans le chemin : $\mu_1(b, b) = \{b, 1, a, 2, c, 3, b\}$ est un circuit.

Application 03 : Chaîne élémentaire et chaîne simple

03.1. Existe-t-il une chaîne élémentaire qui n'est pas simple ?

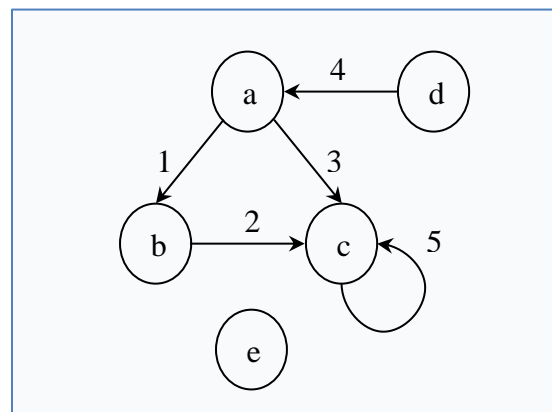
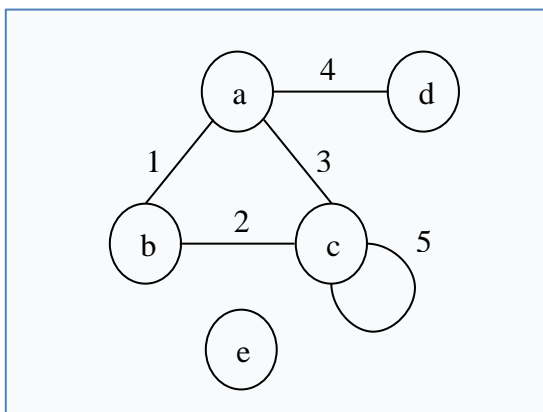
I.3. Mesures dans un graphe

I.3.1. Degré d'un sommet

I.3.1.a. Graphe non orienté

I.3.1.b. Graphe orienté

Exemple 04 : Degrés des sommets



Le degré d'un sommet x est le nombre de liaisons de ce sommet. On le note $d(x)$:

$$\begin{aligned} d : X &\rightarrow \mathbb{N} \\ x &\rightarrow d(x) \end{aligned}$$

$$d(a) = 3$$

$$d(b) = 2$$

$$d(c) = 4$$

La somme des degrés des sommets d'un graphe non orienté est égale au double du nombre d'arêtes.

Les boucles sont comptées deux fois dans le degré d'un sommet (Diestel, 2005).

Nous distinguons deux cas particuliers de sommets :

le sommet pendant d : $d(d) = 1$

et le sommet isolé e : $d(e) = 0$

Pour les graphes orientés, il existe deux degrés:

- degré positif $d^+(x)$: nombre d'arcs sortants du sommet x

- degré négatif $d^-(x)$: nombre d'arcs entrants au sommet x

Par exemple pour le sommet b :

$$d^+(b) = 1 \quad \text{arc } (b, c)$$

$$d^-(b) = 1 \quad \text{arc } (a, b)$$

Finalement, le degré du sommet b est :

$$d(b) = d^+(b) + d^-(b) = 2$$

Nous distinguons deux cas particuliers de sommets dans un graphe orienté :

le sommet source s : (sommet d)

$$d^+(s) > 0 \quad d^-(s) = 0$$

et le sommet puits p :

$$d^+(p) = 0 \quad d^-(p) > 0$$

Application 04 : Etoile du Maroc

- 04.1. Tracer le graphe d'une étoile (drapeau du Maroc): les arcs seront orientés suivant le sens de leur dessin.
- 04.2. Quels sont les degrés (positifs et négatifs) des sommets ?
- 04.3. Y a-t-il une relation entre le nombre d'arcs et les degrés des sommets ?

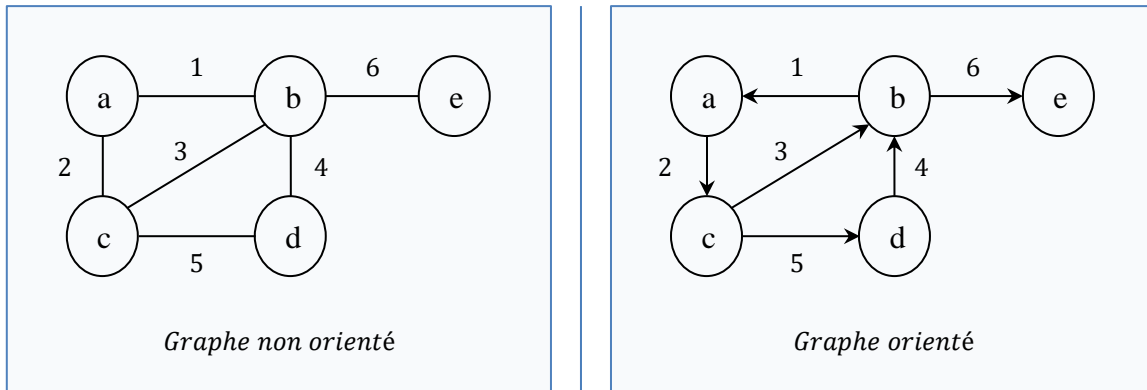
I.3.2. Distances

I.3.2.a. Distance entre deux sommets

La distance entre deux sommets d'un graphe est le nombre minimal d'arcs/arêtes les reliant. C'est la longueur de la plus courte chaîne/ du plus court chemin les reliant.

On note la distance entre deux sommets a et b par : $d(a, b)$.

Exemple 05 : Distances dans un graphe



Dans le graphe non orienté : la distance entre a et e est égale à 2. Il faut au minimum 2 arêtes : 1 et 6. Pour un graphe non orienté : $d(a, e) = d(e, a)$.

Dans le graphe orienté : la distance de a à e est égale à 3. $d(a, e) \neq d(e, a)$.

Le plus court chemin est $\mu^*(a, e) = \{a, 2, c, 3, b, 6, e\}$.
Sa longueur est $l(\mu^*(a, e)) = 3$.

I.3.2.a. Ecartement d'un sommet

L'écartement d'un sommet est la distance maximale existant entre ce sommet et les autres sommets du graphe.

Dans l'Exemple 05:

Dans le graphe non orienté :

Les distances entre le sommet a et les autres sommets b, c, d, e , sont respectivement : 1, 1, 2, 2. Par conséquent, l'écartement est : $\rho(a) = 2$.

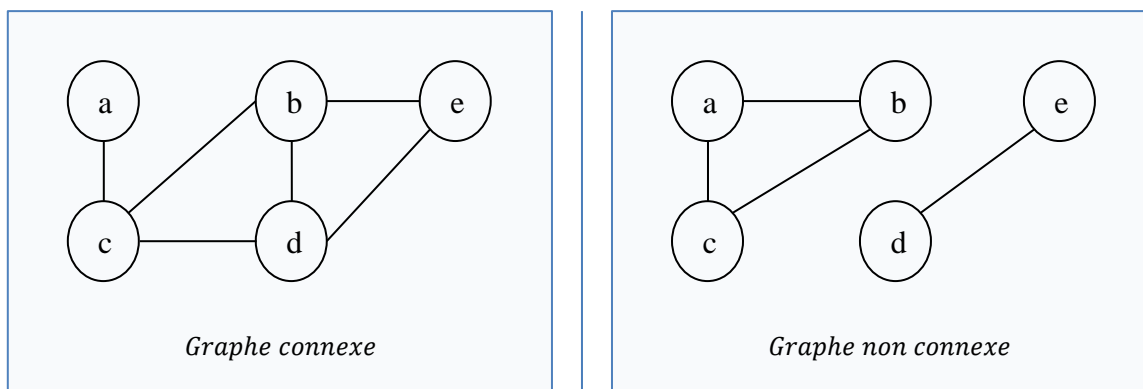
Dans le graphe orienté :

Les distances entre le sommet a et les autres sommets b, c, d, e , sont respectivement : 2, 1, 2, 3. Par conséquent, $\rho(a) = 3$.

Application 05 : Ecartement dans un graphe partiel

- 05.1. Comparer l'écartement d'un même sommet x dans un graphe non orienté connexe G et dans G_1 un graphe partiel connexe de G .
- 05.2. Comparer la distance entre 2 sommets a et b dans G et dans G_2 un graphe partiel de G .
- 05.3. Dessiner un graphe orienté G^o d'ordre 4 dans lequel : l'écartement de chaque sommet est égal au minimum des distances entre ce sommet et les autres.

Exemple 06 : Ecartements dans un graphe non connexe



Le graphe à droite est non connexe (ses sommets ne sont pas tous connectés), il nous est donc impossible d'aller d'un sommet et arriver à tous les autres.

Les écartements sont infinis. Par exemple : $\rho(a) = \infty$.

I.3.2.a. Centre et rayon d'un graphe

Le centre d'un graphe est le sommet d'écartement minimal. Le centre n'est pas nécessairement unique.

On appelle : rayon d'un graphe G l'écartement de l'un de ses centres. Il est noté : $\rho(G)$.

Par exemple, dans le graphe connexe de l'*Exemple 06*, les écartements des sommets sont :

$\rho(a) = 3$	$\max(2, 1, 2, 3)$
$\rho(b) = 2$	$\max(2, 1, 1, 1)$
$\rho(c) = 2$	$\max(1, 1, 1, 2)$
$\rho(d) = 2$	$\max(2, 1, 1, 1)$
$\rho(e) = 3$	$\max(3, 1, 2, 1)$

Donc, les centres du graphe sont les sommets : b, c, d .

I.3.2.a. Diamètre d'un graphe



On appelle diamètre d'un graphe G la distance maximale entre deux sommets de G .

Il est noté $\delta(G)$.



Le diamètre d'un graphe n'est pas forcément égal au double de son rayon.

Application 06 : Diamètre et rayon d'un graphe

- 06.1. Dessiner un graphe non orienté G d'ordre 6 ayant un seul centre dont l'écartement est 1.
- 06.2. Quels sont les écartements des autres sommets?
- 06.3. Quel est le nombre d'arêtes minimal de G pour avoir 2 centres dont l'écartement est 1?
- 06.4. Dessiner un graphe non orienté G_2 d'ordre 4 tel que : $\delta(G_2) = \rho(G_2)$
- 06.5. Ajouter/ retirer une arête de G_2 pour avoir : $\delta(G_2) = 2 \rho(G_2)$.

I.4. Relations - graphe



La partie la plus délicate en théorie des graphes est la représentation sagittale du graphe. Sa conception repose essentiellement sur la notion de relations binaires ; pour l'expliquer, nous allons étudier deux exemples de la vie courante : un pour les garçons, et un autre pour les filles :

Exemple 07 : Facebook

5 filles viennent de créer des comptes sur Facebook :

- Maha : amie avec Fati et Salma
- Fati : amie avec Maha et Salma
- Salma : amie avec Maha, Fati, Imane
- Imane : amie avec Karima et Salma
- Karima : amie avec Imane

Exemple 08 : Transferts FIFA

En 2017, 5 joueurs de 4 clubs ont signé les contrats suivants :

- Neymar : transfert : Barça \rightarrow PSG
- Matuidi : transfert : PSG \rightarrow Juve
- Dani Alves : Juve \rightarrow PSG
- Rodriguez: R. Madid \rightarrow PSG
- Messi : renouvellement au Barça

La première étape consiste à définir les ensembles étudiés puis à les mettre en relation :

Exemple 07:

- **Ensembles** : - fille,
 - ses amies
- **Relation** : lien d'amitié.

Exemple 08:

- **Ensembles** : - club de départ,
 - club d'arrivée.
- **Relation** : transfert du joueur
(renouvellement inclus).



Cette partie fait intervenir les notions étudiées sur les ensembles et celles sur les applications et fonctions.

Dans l'*Exemple 07*, il s'agit du même ensemble de départ et d'arrivée, et la relation est bijective (si une fille A est amie avec une deuxième fille B, alors B est aussi amie avec A). Cela nous rappelle les arcs doubles définis en -I.1.- pour les graphes non orientés.

Par contre, dans l'*Exemple 08*, les relations traduisent le sens du transfert du joueur : du club de départ vers le club d'arrivée.

Il s'agit donc d'un graphe non orienté dans l'*Exemple 07*, et d'un graphe orienté dans l'*Exemple 08*. Nous résumerons les données des deux cas dans les tableaux suivants:

Exemple 07:

La première colonne du tableau comportera l'ensemble des filles, et la première ligne comportera leurs amies. Nous retrouvons ainsi les mêmes éléments, et la relation sera traduite par un signe \times par exemple :

	Maha	Fati	Salma	Imane	Karima
Maha		\times	\times		
Fati	\times		\times		
Salma	\times	\times		\times	
Imane			\times		\times
Karima				\times	



Ici, la flèche n'est pas nécessaire et le tableau est symétrique, c'est une caractéristique des graphes non orientés.

Exemple 08:

Le premier ensemble des clubs de départ figurera sur la première colonne et l'ensemble des clubs d'arrivée figurera sur la première ligne, mais la relation ici comporte le nom du joueur, il remplacera donc le signe \times :



	Barça	PSG	Juve
Barça	Messi	Neymar	
PSG			Matuidi
Juve		D. Alves	
Madrid		Rodriguez	



Le sens de la flèche du tableau est très important : il détermine le sens du transfert, l'invertir signifie que le joueur est passé du club d'arrivée vers son club d'origine.

Exemple 09 : Matières enseignées à l'ESI

Dans l'*Exemple 08*, la relation contient une information importante : le nom du joueur. Elle varie d'un transfert à un autre. Il existe un autre type de représentations classique où la relation lie deux ensembles différents, mais elle ne comporte pas d'information. Par exemple : les matières enseignées à l'ESI durant 4 semestres :

La relation lie deux ensembles : les matières et les semestres. Elle sert uniquement à exprimer cette liaison par la phrase :

une matière est en relation avec les semestres où elle est enseignée

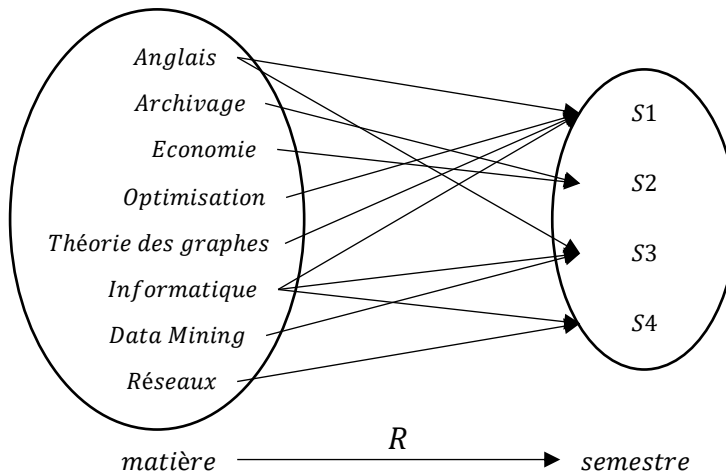
S1	S2	S3	S4
<i>Théorie des graphes</i>	<i>Economie</i>	<i>Data Mining</i>	<i>Informatique</i>
<i>Informatique</i>	<i>Archivage</i>	<i>Informatique</i>	<i>Réseaux</i>
<i>Anglais</i>		<i>Anglais</i>	
<i>Optimisation</i>			

Si nous considérons de plus le nom du professeur qui enseigne la matière, la relation comportera cette information aussi (comme dans l'*Exemple 08*).

I.4.1. Relations binaires

Pour mieux expliquer la notion de relation, nous allons reprendre l'*Exemple 09*:

Nous représentons les ensembles par des diagrammes de Venn, puis nous lions les éléments en relation par des flèches :



Notons : E l'ensemble de départ, F l'ensemble d'arrivée et $U \subset E \times F$ l'ensemble des arcs.

Une relation R de E vers F est donnée par le triplet d'ensembles (E, F, U)

Un élément x de E est en relation avec un élément y de F si $(x, y) \in U$. On note xRy .

x est un prédécesseur de y, il est appelé origine de (x, y)

De même, y est un successeur de x, il est appelé extrémité de (x, y)

L'ensemble des prédécesseurs de y est noté $\Gamma^-(y)$

Et l'ensemble des successeurs de x est noté $\Gamma^+(x)$

$d^-(y) = \text{Card}\Gamma^-(y)$ est le degré entrant en y

$d^+(x) = \text{Card}\Gamma^+(x)$ est le degré sortant de x

Le domaine de la relation R est : $D_R = \{x \in E \mid \exists y \in F, (x, y) \in U\}$

L'image de la relation R est : $\text{Im}_R = \{y \in F \mid \exists x \in E, (x, y) \in U\}$

Application 07 : Relation des matières enseignées

Nous définissons les ensembles :

$$E = \{TdG, Info, Ang, Opti, Eco, Arch, DM, Res, Big Data, BI\}$$

$$F = \{S1, S2, S3, S4, S5, S6\}$$

A partir de l'**Exemple 09 : Matières enseignées à l'ESI**, déterminer :

07.1. le domaine D_R

07.2. l'image : Im_R

07.3. $\Gamma^+(Informatique)$

07.4. $\Gamma^-(S1)$

07.5. $d^+(Economie)$



Ces dernières définitions nous montrent que les fonctions et les applications ne sont en fait que des cas particuliers de relations.

Nous avons pu constater dans l'**Exemple 08** qu'il nous est impossible de modéliser les liens en utilisant les applications et les fonctions classiques de l'ensemble de départ vers celui d'arrivée. D'où le besoin d'une notion plus générale : les relations des graphes.

I.4.2. Matrice d'adjacence

Après avoir schématisé la relation dans un graphe, il nous reste à trouver une représentation qui permet de faire des calculs : la représentation matricielle.

Notons : $E = \{x_1, x_2, \dots, x_p\}$ un ensemble de départ,

$F = \{y_1, y_2, \dots, y_n\}$ un ensemble d'arrivée,

et $R = (E, F, U)$ une relation

Nous appellerons matrice d'adjacence de la relation R la matrice booléenne $M_R \in M_{p,n}(B)$

telle que : $M_R = (m_{i,j})$ avec $m_{i,j} = \begin{cases} 1 & \text{si } (x_i, y_j) \in U \\ 0 & \text{sinon} \end{cases}$



Une matrice d'adjacence M_R est booléenne : comporte uniquement des 0 et des 1.

Une matrice valuée (**Exemple 08 : Transferts FIFA**) peut contenir d'autres valeurs.

M_R est rarement considérée dans $M_{p,n}(Z)$: quand le nombre d'arcs identiques est significatif.

Dans la matrice d'adjacence M_R , les lignes correspondent aux éléments de l'ensemble de départ et les colonnes correspondent aux éléments de l'ensemble d'arrivée.

I.4.3. Analogie avec les applications

Fonction

Toute fonction $f : E \rightarrow F$ est une relation

$$f(x) = y \Rightarrow x R y$$

Une relation R est une fonction *ssi* chaque élément x de E , a au plus un successeur :

$$\forall x \in E : \text{Card}(\Gamma^+(x)) \leq 1 \Leftrightarrow \forall x \in E : d^+(x) \leq 1$$

Application

Une relation $R : E \rightarrow F$ est une application *ssi* :

R est une fonction

Son domaine de définition est égal à E :

$$\forall x \in E : \text{Card}(\Gamma^+(x)) = 1 \Leftrightarrow \forall x \in E : d^+(x) = 1$$

f est dite injective *ssi* chaque élément de F a au plus un prédécesseur

$$\forall y \in F : \text{Card}(\Gamma^-(y)) \leq 1 \Leftrightarrow \forall y \in F : d^-(y) \leq 1$$

$$\text{Dans ce cas : } \text{Card}(E) \leq \text{Card}(F)$$

f est dite surjective *ssi* chaque élément de F a au moins un prédécesseur

$$\forall y \in F : \text{Card}(\Gamma^-(y)) \geq 1 \Leftrightarrow \forall y \in F : d^-(y) \geq 1$$

$$\text{Dans ce cas : } \text{Card}(E) \geq \text{Card}(F)$$

f est bijective *ssi* elle est à la fois injective et surjective

$$\forall y \in F : \text{Card}(\Gamma^-(y)) = 1 \Leftrightarrow \forall y \in F : d^-(y) = 1$$

$$\text{Dans ce cas : } \text{Card}(E) = \text{Card}(F)$$

Application 08 : Diagrammes de Venn

08.1. Représenter à l'aide de diagrammes de Venn :

08.1.1. Une relation qui n'est pas fonction

08.1.2. Une fonction qui n'est pas application

08.1.3. Une application injective et non surjective

08.2. Représenter sur des diagrammes de Venn les relations des [exemples](#) *Exemple*

07 et Exemple 08

I.4.4. Opérations sur les relations

Les définitions des relations nous permettent d'étudier un ensemble d'opérations sur les graphes :

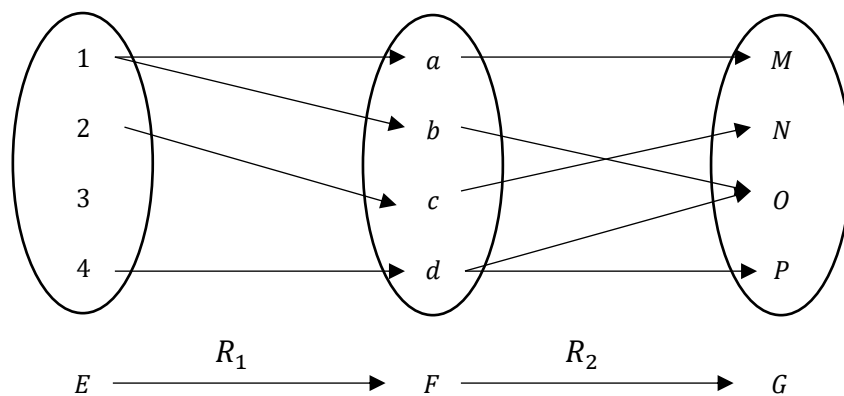
I.4.4.a. Composition de relations

Soient $R_1 = (E, F, U_1)$ et $R_2 = (F, G, U_2)$ deux relations,

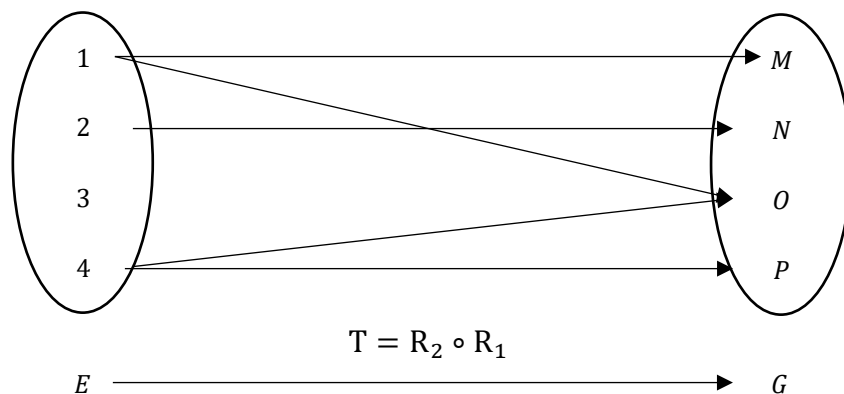
La composition des relations s'écrit : $T = R_2 \circ R_1 = (E, G, U)$, et ses arcs sont :


$$U = \{(x, z) \in E \times G \mid \exists y \in F, (x, y) \in U_1 \text{ et } (y, z) \in U_2\}$$

Prenons l'exemple des deux relations suivantes :



La composition $T = R_2 \circ R_1$ nous donne :



La matrice d'adjacence de la composition des deux relations correspond au produit matriciel des deux matrices d'adjacence (produit en algèbre de Boole binaire). 

$$M_{R_2 \circ R_1} = M_{R_1} \times M_{R_2}$$

En effet, dans la matrice $M_{R_2 \circ R_1}$:

$m_{i,j} = 1$ signifie qu'il existe un k pour lequel : $m_{i,k} \times m_{k,j} = 1$. Donc : $m_{i,k} = 1$ et $m_{k,j} = 1$, c'est-à-dire qu'il existe un arc (x_i, z_k) dans R_1 et un arc (z_k, y_j) dans R_2 .



En algèbre de Boole binaire :

$$\begin{array}{llll} 0 + 0 = 0, & 0 + 1 = 1, & 1 + 0 = 1, & \text{et} \quad 1 + 1 = 1 \\ 0 \times 0 = 0, & 0 \times 1 = 0, & 1 \times 0 = 0, & 1 \times 1 = 1 \end{array}$$

Attention à l'ordre de la composition.

Application 09 : Matrice d'adjacence d'une composition

09.1. Retrouver la matrice d'adjacence de la composition (exemple précédent) de deux façons différentes.

I.4.4.b. Inverse d'une relation

Soit $R = (E, F, U)$ une relation.

La relation réciproque de R , est notée $R^{-1} = (F, E, U')$, et ses arcs sont définis par :

$$U' = \{(y, x) \in F \times E \mid (x, y) \in U\}$$

$$M_{R^{-1}} = {}^t M_R$$

I.4.5. Relations dans un ensemble


Dans l'*Exemple 07*, l'ensemble de départ est exactement celui d'arrivée. Nous parlons alors de relation dans un ensemble. Nous pouvons représenter ce type de relations par une matrice d'adjacence ou par des diagrammes sagittaux.

I.4.5.a. Propriétés d'une relation

Notons R une relation dans un ensemble E .

On dit que R est :

- Réflexive si : $\forall x \in E \quad xRx$
tout élément de S possède une boucle
 $M = M + Id_n$
- Symétrique si : $\forall x, y \in E \quad xRy \Leftrightarrow yRx$
tous les arcs sont doubles
 $M = {}^t M$

- Antisymétrique si : $\forall x, y \in E \quad xRy \text{ et } yRx \Rightarrow x = y$ 
aucun arc n'est double
 $M * {}^tM + Id_n = Id_n$ (produit terme à terme)
- Transitive si : $\forall x, y, z \in E \quad xRy \text{ et } yRz \Rightarrow xRz$
pour chaque couple d'arcs adjacents, l'arc résultant est aussi un arc du graphe.
 $M^2 + M = M$



Les propriétés symétrique et antisymétrique ne sont pas complémentaires.

Application 10 : Graphes des 4 propriétés

- 10.1. Pour chacune des 4 propriétés précédentes, dessiner un graphe d'ordre 4 dont la relation vérifie la propriété.
- 10.2. Dessiner un graphe d'ordre 4 tel que :
 - la relation est réflexive et symétrique,
 - la somme des éléments de la matrice d'adjacence est le double de la somme de ses éléments diagonaux.

I.4.5.b. Relation d'équivalence

Soit $R = (E, E, U)$ une relation dans un ensemble E

R est une relation d'équivalence ssi elle est :

- Réflexive,
- Symétrique,
- Transitive.

Le sous ensemble de E : $Cl(x) = \{y \in E \mid xRy\}$ est appelé : classe d'équivalence de x .

L'ensemble des classes d'équivalences de R est appelé : ensemble quotient de E par R , on le note : E/R . Il forme une partition de E :

$$\forall x \in E, Cl(x) \neq \emptyset \quad (Cl(x) \text{ contient } x).$$

$$\forall (x, y) \in E^2, xRy \Leftrightarrow Cl(x) = Cl(y)$$

$$\forall x \in E, \bigcup_{x \in E} Cl(x) = E$$

Application 11 : Relation d'équivalence

11.1. Notons a un élément de l'intervalle $I \subset \mathbb{R}$

Montrer que la relation : $f \sim g$ est bien une relation d'équivalence dans l'ensemble des applications $E = \{f : I \rightarrow \mathbb{R}\}$.

I.4.5.c. Relation d'ordre

Soit $R = (E, E, U)$ une relation dans un ensemble E

R est une relation d'ordre ssi elle est :

- Réflexive,
- Antisymétrique, (différence par rapport à la relation d'équivalence)
- Transitive.

L'ordre est total si : $\forall (x, y) \in E^2, xRy$ ou yRx , et partiel sinon.

Les relations d'ordre sont plus simples à représenter, nous utilisons pour cela le Diagramme de Hasse :

- Nous représentons d'abord les sommets (sens croissant)
- Ensuite, nous représentons les boucles (réflexivité)

les raccourcis sont implicites (transitivité) et le graphe est non orienté (antisymétrie)

par convention : la relation se lit de bas en haut ou de gauche à droite

Exemple 10 : Parties d'un ensemble

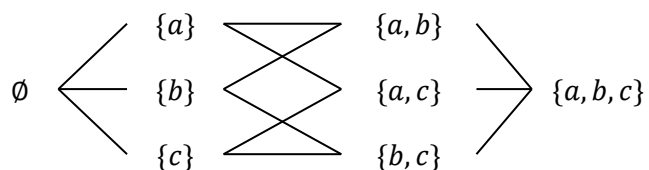
La relation \subset sur les parties d'un ensemble $P(E)$ est une relation d'ordre

Nous avons :

- $\forall A \subset E, A \subset A$
- $\forall A, B \subset E, (A \subset B \text{ et } B \subset A) \Rightarrow A = B$
- $\forall A, B, C \subset E, (A \subset B \text{ et } B \subset C) \Rightarrow A \subset C$

Nous prenons l'exemple de $P(\{a, b, c\})$:

Le diagramme de Hasse :



Nous remarquons que la relation n'est pas d'ordre total.

Application 12 : Relation d'ordre

- 12.1. Considérons dans l'ensemble N l'intervalle $[[1,5]]$
Montrer que la relation : \leq est une relation d'ordre total sur $[[1,5]]$.
- 12.2. Est-ce que la relation $<$ est également une relation d'ordre ? justifier.

Application 13 : Relation de division sur N^*

- 13.1. Nous considérons la relation R définie sur l'ensemble N^* par:
 xRy : x est un diviseur de y
- 13.2. Montrer que la relation R est transitive.
- 13.3. R est-elle une relation d'ordre ?
- 13.4. Dessiner un graphe orienté dont les sommets sont les entiers compris entre 1 et 12 et dont les arcs représentent la relation R .
- 13.5. Dessiner un graphe orienté sur le même ensemble dont la relation R' est définie par : $xR'y \Leftrightarrow yRx$
- 13.6. Est-ce que la relation R'' définie par : $xR''y \Leftrightarrow (xRy \text{ ou } xR'y)$ est une relation d'ordre ?

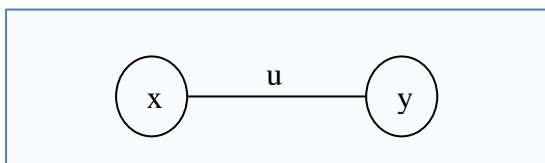
I.5. Représentation matricielle

I.5.1. Définitions

I.5.1.a. Graphe non orienté

Deux sommets x et y sont adjacents s'il existe au moins une arête u qui les lie.

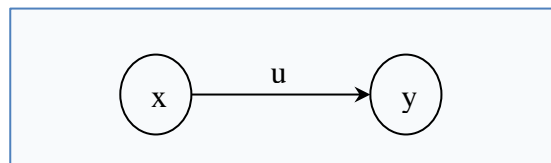
x et y sont alors appelés extrémités terminales de l'arête u :



I.5.1.b. Graphe orienté

Pour un graphe orienté, la notion d'adjacence reste valable.

Nous distinguons toutefois l'extrémité initiale x et l'extrémité finale y :



La notion d'incidence quant à elle, lie les sommets aux arcs. Elle exprime le sens de l'arc par rapport au sommet.

I.5.2. Matrice d'adjacence

La matrice d'adjacence d'un graphe est semblable à celle vue précédemment pour les relations :

Notons $G = (X, U)$ un graphe d'ordre n , et $x_1, x_2 \dots x_n$ ses sommets .

La matrice d'adjacence de G , notée M_G , est définie sur $M_n(B)$ par :

$$M_G = (m_{i,j}) \quad \text{avec} \quad m_{i,j} = \begin{cases} 1 & \text{si } (x_i, x_j) \in U \\ 0 & \text{sinon} \end{cases} \quad \triangle$$



Pour un graphe non orienté : $(x_i, x_j) = (x_j, x_i)$

Rarement, quand le nombre d'arcs/arêtes identiques est significatif :

M_G est considérée dans $M_n(\mathbb{Z})$.

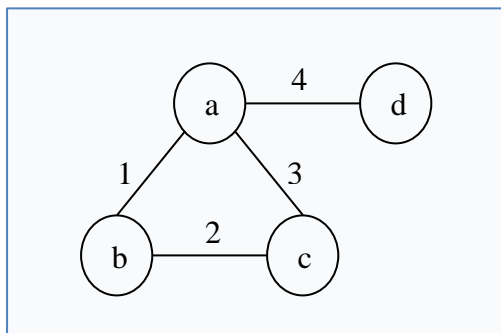
Pour un graphe pondéré I.8, M_G peut contenir d'autres valeurs.

Suivant la nature du graphe (orienté ou non orienté), nous distinguons deux types de matrices :

I.5.2.a. Graphe non orienté

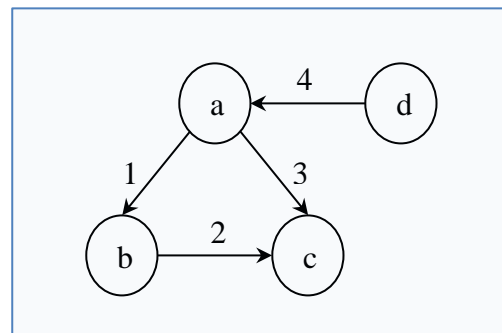
I.5.2.b. Graphe orienté

Exemple 11 : Matrice d'adjacence d'un graphe



La matrice d'adjacence non orientée M_G :

$$\begin{array}{c} \begin{matrix} & a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} \end{array} \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$



La matrice d'adjacence orientée M_{G^o} :

$$\begin{array}{c} \begin{matrix} & a & b & c & d \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} \end{array} \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

Application 14 : Symétrie des matrices d'adjacence

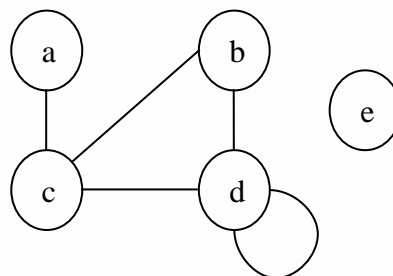
- 14.1. Montrer qu'une matrice d'adjacence non orientée est symétrique.
- 14.2. Donner une CNS pour qu'une matrice d'adjacence orientée soit symétrique.
- 14.3. A quoi correspond la somme des valeurs d'une ligne de la matrice d'adjacence non orientée ?
- 14.4. A quoi correspond la somme des éléments d'une matrice d'adjacence ?
- 14.5. Pour une matrice d'adjacence orientée, que représente la somme des valeurs :
 - d'une ligne ?
 - d'une colonne ?

Application 15 : Interprétation des matrices d'adjacence

- 15.1. Donner les matrices d'adjacence des *exemples Exemple 07, Exemple 08 et Exemple 09*.
- 15.2. Quelle relation y a-t-il entre la somme des éléments de la matrice d'adjacence et les liens d'amitié de l'*Exemple 07*?
- 15.3. Que représentent les éléments sur la diagonale de la matrice d'adjacence de l'*Exemple 08*?
- 15.4. Quel effet l'inversion des ensembles de l'*Exemple 09* a-t-elle sur la matrice d'adjacence?

Application 16 : Le carré d'une matrice d'adjacence

- 16.1. Donner la matrice d'adjacence M_G du graphe suivant :



- 16.2. Calculer M_G^2 .
- 16.3. Dessiner le graphe défini par M_G^2

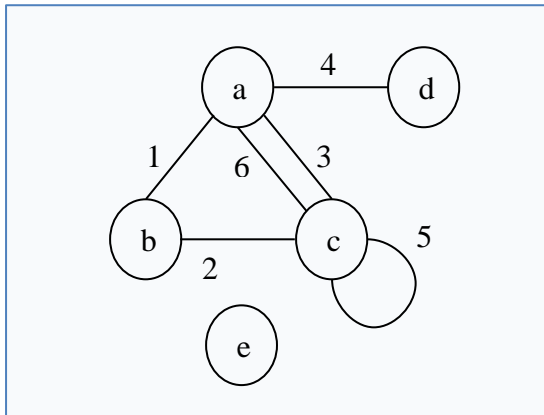
I.5.3. Matrice d'incidence

La matrice d'incidence d'un graphe G à n sommets et m arcs est de taille $n \times m$.

Suivant le type du graphe, nous distinguons :

Exemple 12 : Matrice d'incidence d'un graphe

I.5.3.a. Graphe non orienté



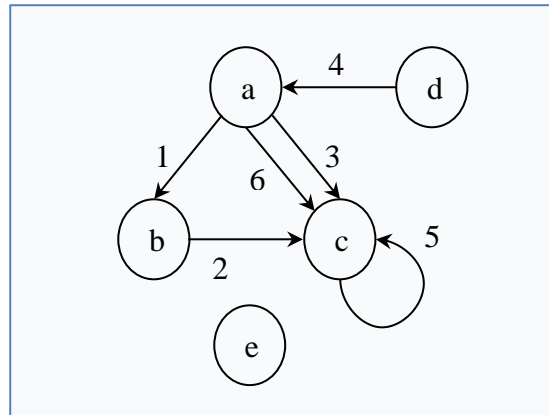
La matrice d'incidence est M_i :

$$\begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \\ e \end{matrix} & \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 2 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix}$$

Le coefficient (i, j) vaut :

- 1 si : s_i est une extrémité de u_j
- 2 si : l'arête u_j est une boucle sur s_i
- 0 sinon

I.5.3.b. Graphe orienté



La matrice d'incidence est M_i^o :

$$\begin{matrix} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \\ e \end{matrix} & \begin{pmatrix} -1 & 0 & -1 & 1 & 0 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{matrix}$$

Le coefficient (i, j) vaut :

- 1 si : l'arc u_j sort du sommet s_i
- 1 si : l'arc u_j entre dans le sommet s_i
- 0 sinon

Application 17 : Matrices d'adjacence et d'incidence

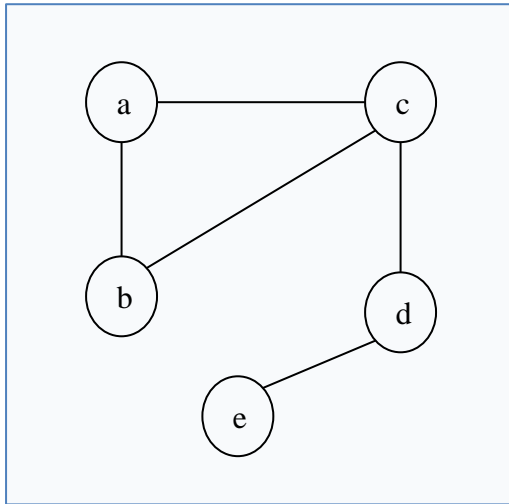
- 17.1. Donner les matrices d'adjacence de l'Exemple 12.
- 17.2. Peut-on construire un graphe d'ordre 5 tel que ses matrices d'adjacence et d'incidence soient identiques ?

I.5.4. Matrice laplacienne

Il existe une troisième représentation matricielle d'un graphe : la matrice laplacienne ou matrice de Laplace, notée M_L .

Elle regroupe à la fois les degrés d'un graphe et l'adjacence de ses arcs/arêtes :

Exemple 13 : Matrice laplacienne d'un graphe



La matrice des degrés du graphe est :

$$M_d = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

La matrice d'adjacence du graphe est :

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

La matrice laplacienne du graphe est définie par : $M_L = (m_{i,j})_{i,j}$ telle que :

$$m_{i,j} = \begin{cases} \deg(x_i) & \text{si : } i = j \\ -1 & \text{si : } i \neq j \text{ et } (x_i, x_j) \in U \\ 0 & \text{sinon} \end{cases}$$

Elle est donc égale à : $M_d - M_G$.

La matrice laplacienne est utilisée par le théorème de Kirchhoff pour calculer le nombre d'II.3 d'un graphe.

En théorie spectrale des graphes, le spectre d'une matrice laplacienne permet de partitionner un graphe.

Application 18 : Matrice laplacienne

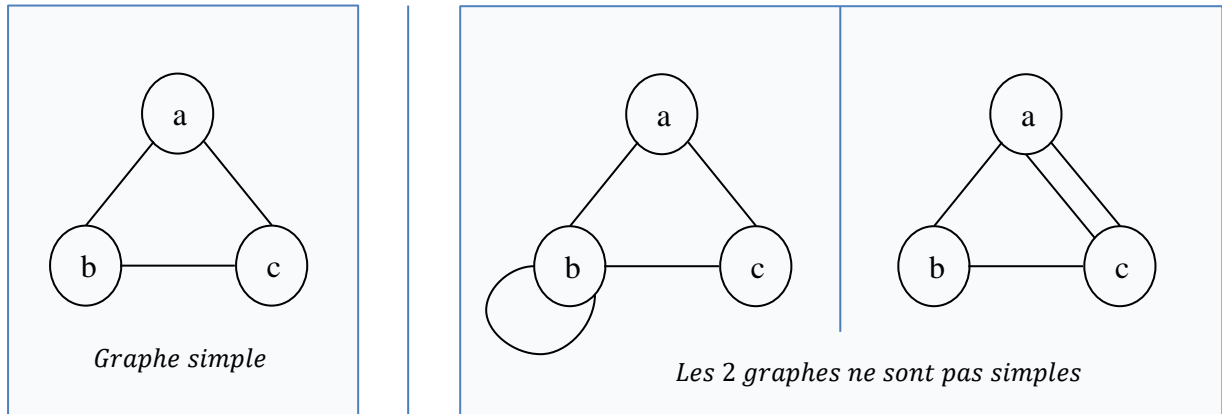
18.1. Donner les matrices laplaciennes de l'Exemple 12.

I.6. Graphes particuliers

I.6.1. Graphe simple

Un graphe simple est un graphe sans boucle et sans arêtes parallèles.

Exemple 14 : Graphe simple



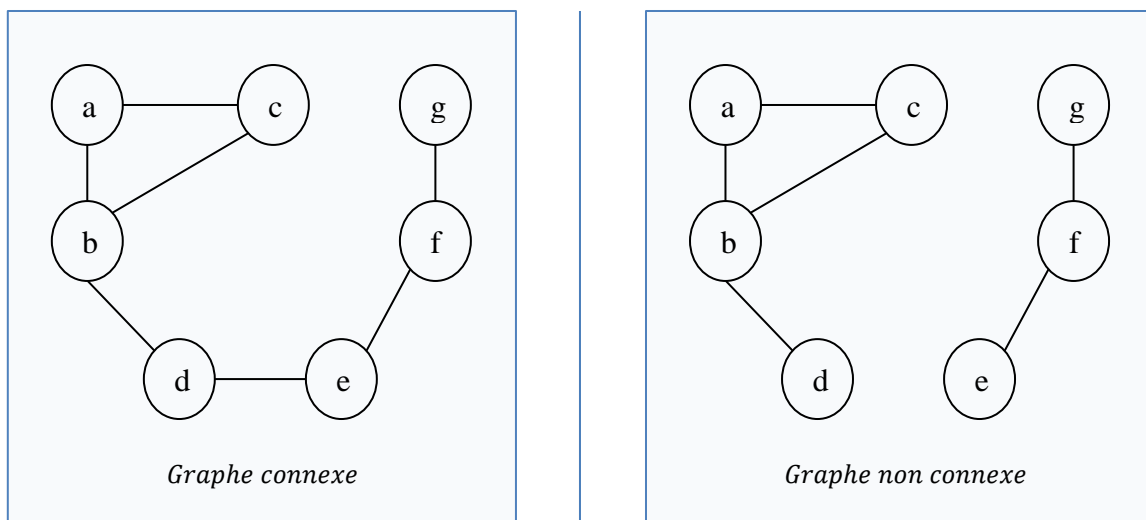
Les deux graphes à droite ne sont pas simples parce que le premier contient une boucle et le second deux arêtes parallèles.

I.6.2. Graphe connexe

Un graphe est connexe s'il est possible, à partir de n'importe quel sommet, d'arriver à tous les autres en suivant les arcs.

Un graphe non connexe peut être décomposé en composantes connexes.

Exemple 15 : Graphe connexe



Dans le graphe à droite, les composantes connexes sont $\{a, b, c, d\}$ et $\{g, f, e\}$.

Un graphe est h -connexe s'il reste connexe en enlevant $(h - 1)$ arêtes quelconques.

Un point d'articulation est un sommet dont la suppression disconnecte le graphe.

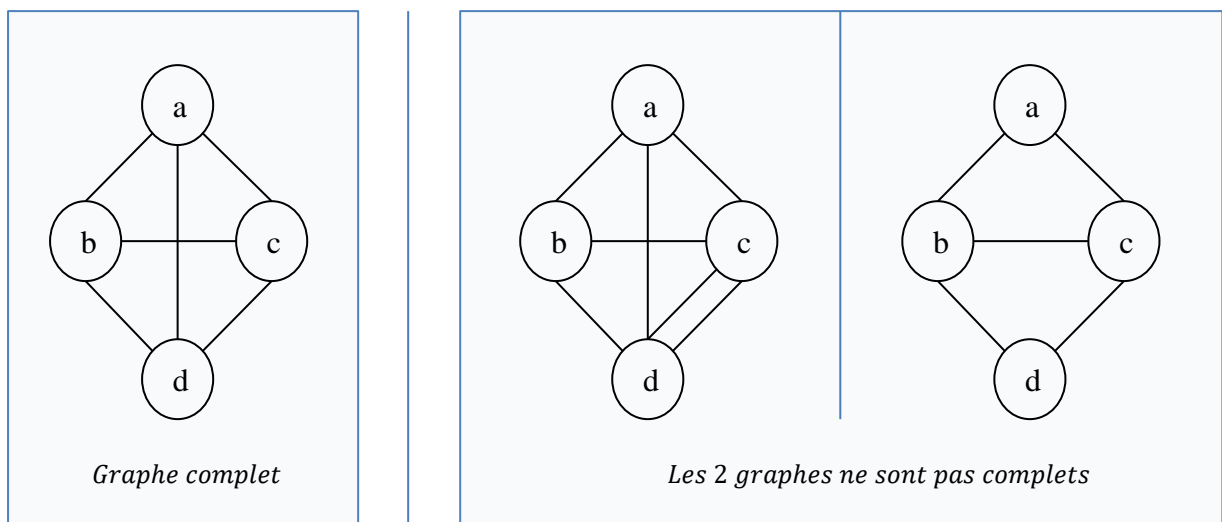
Un isthme est un arc/ arête dont la suppression disconnecte le graphe.

Un ensemble d'articulation est un ensemble de sommets dont la suppression disconnecte le graphe. Le cardinal minimal d'un ensemble d'articulation est appelé connectivité du graphe.

I.6.3. Graphe complet

Un graphe est complet s'il est simple et chacun de ses sommets est lié à tous les autres sommets.

Exemple 16 : Graphe complet



Le premier des deux graphes à droite n'est pas complet parce qu'il n'est pas simple, et le second parce que les sommets a et d ne sont pas adjacents.

Le sous-graphe $G_2 = (X_2, U_2)$ avec : $X_2 = \{b, c, d\}$ et $U_2 = \{(b, c), (b, d), (c, d)\}$ est complet, on l'appelle une clique.

I.6.4. Graphe biparti

Un graphe est dit biparti si l'ensemble de sommets peut être partitionné en deux classes de sorte que les sommets d'une même classe ne soient jamais adjacents.

Les graphes de fonctions, applications, bijections sont des graphes bipartis.

On définit aussi des graphes bipartis complets, notés $K_{m,n}$

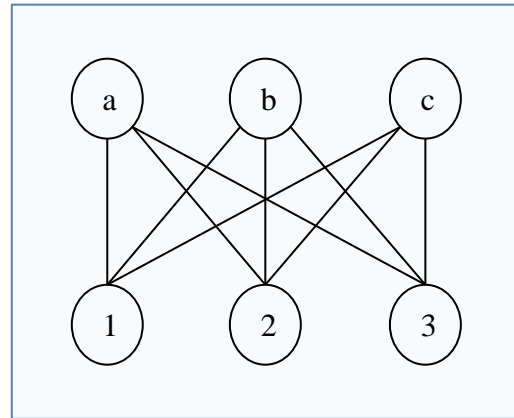
Exemple 17 : Graphe biparti

Le graphe ci-contre est biparti :

L'ensemble des sommets est partitionné en :

$\{a, b, c\}$ et $\{1, 2, 3\}$.

De plus, il est complet : $K_{3,3}$



Application 19 : Graphe biparti

19.1. Montrer que :

Un graphe est biparti si et seulement s'il ne contient aucun cycle de longueur impaire?

I.6.5. Graphe planaire

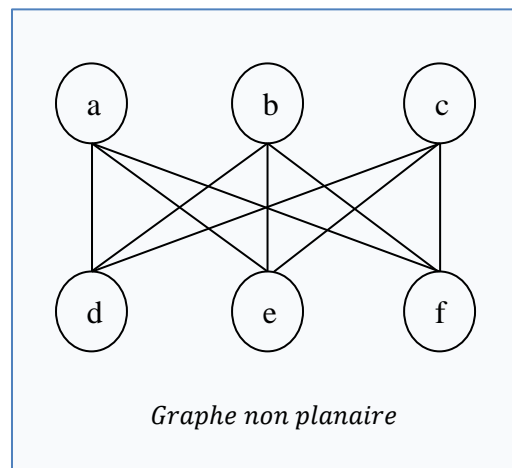
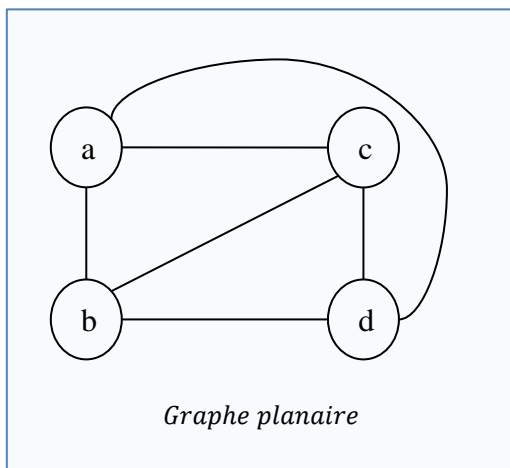


Un Graphe est dit planaire s'il admet une représentation sagittale planaire :

Les sommets d'un graphe planaire sont tous sur le même plan et les arêtes ne se croisent pas.

Dans la représentation planaire d'un graphe, on appelle face toute région (finie ou infinie) du plan non traversée par des arêtes.

Exemple 18 : Graphe planaire



Pour qu'un graphe soit planaire :

Il faut qu'il admette au moins une représentation planaire, pas que toutes le soient.

Dans le graphe à gauche de l'*Exemple 18*, il y a 4 faces:

- délimitée par les sommets a, c, b
- délimitée par les sommets c, b, d
- délimitée par les sommets d, c, a
- délimitée par les sommets a, b, d (infinie)

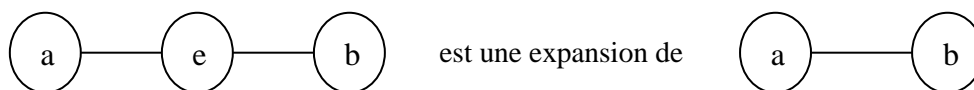
Ce même graphe admet une autre représentation sagittale : l'arête (a, d) croise l'arête (b, c), mais il a suffi de trouver une représentation planaire pour dire que le graphe est planaire. Une telle représentation est impossible pour le graphe à droite, c'est pourquoi il est non planaire.

Caractérisation de Kuratowski :

Un graphe fini est planaire si et seulement s'il ne contient pas de sous-graphe qui est une expansion de K_5 (clique à 5 sommets) ou de $K_{3,3}$ (graphe complet biparti à 3 + 3 sommets).

L'expansion d'un graphe est le résultat de l'ajout d'un ou de plusieurs sommets sur un ou plusieurs l'arêtes.

Par exemple :



Formule d'Euler :

Dans un graphe planaire d'ordre n avec m arêtes, le nombre de faces est :

$$f = m - n + 2$$

Application 20 : Le cube

- 20.1. Montrer que le cube admet une représentation planaire en utilisant la caractérisation de Kuratowski.
- 20.2. Donner une représentation planaire d'un cube.
- 20.3. Retrouver le nombre de faces du cube en utilisant la formule d'Euler.

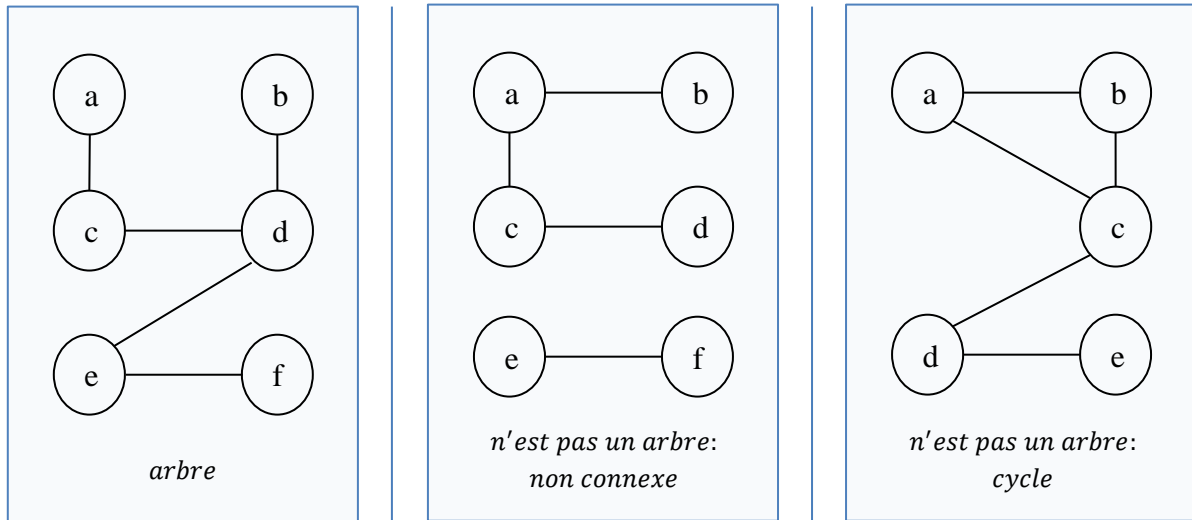
I.6.6. Arbre

I.6.6.a. Arbre non orienté

Un graphe non orienté G est appelé arbre s'il est à la fois connexe et sans cycle.

Les sommets d'un arbre sont appelés : feuilles.

Exemple 19 : Arbre



Si on rajoute une arête u à G :

- soit le nombre de composantes connexes diminue de 1 :
 u lie dans ce cas deux composantes connexes pour former une nouvelle.
- soit le nombre de composantes connexes reste inchangé
 u appartient à un cycle du nouveau graphe, puisqu'il relie deux sommets appartenant à la même composante connexe.

Application 21 : Arêtes d'un arbre

21.1. Montrer les propriétés suivantes :

- Un graphe connexe d'ordre n doit posséder au moins $(n - 1)$ arêtes.
- Un graphe sans cycle d'ordre n possède au plus $(n - 1)$ arêtes.
- Un arbre possède exactement $(n - 1)$ arêtes.

Les propositions suivantes sont équivalentes:

- G est un arbre,
- G est connexe et sans cycle,
- G est sans cycle avec $(n - 1)$ arêtes,
- G est sans cycle, et lui ajouter une arête forme un seul cycle élémentaire,
- G est connexe avec $(n - 1)$ arêtes,
- G est connexe, et lui retirer une arête le rend non connexe,
- Tout couple de sommets du graphe est relié par une chaîne unique.

Application 22 : Sommets pendants d'un arbre

22.1. Montrer que:
tout arbre fini d'ordre ≥ 2 comporte au moins deux sommets pendants.

Un graphe dont les composantes connexes sont des arbres est appelé : forêt.

Une forêt sur n sommets avec p composantes connexes possède $(n - p)$ arêtes.

Dans l'*Exemple 19*, le graphe au milieu est une forêt.

On appelle arbre couvrant (ou arbre maximal) un graphe partiel qui est aussi un arbre.

Codage de Prüfer :

Le codage de Prüfer est une manière très compacte de décrire un arbre G dont les sommets sont numérotés $X = \{1, 2, \dots, n\}$. Il est utilisé pour coder ou décoder un arbre (Müller, 2012) :

Codage :

Tant qu'il reste plus de deux sommets dans l'arbre G

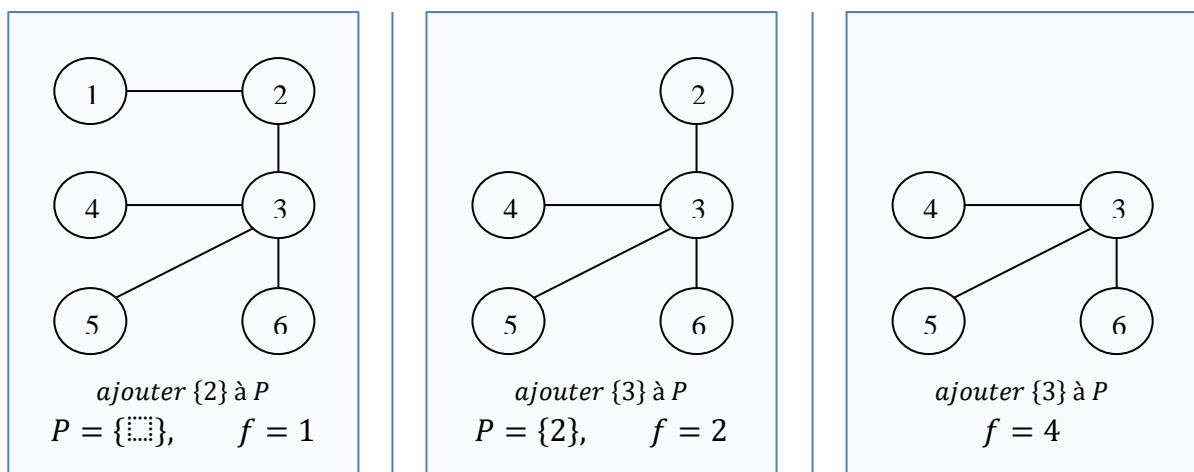
Identifier la feuille f de l'arbre ayant le numéro minimal

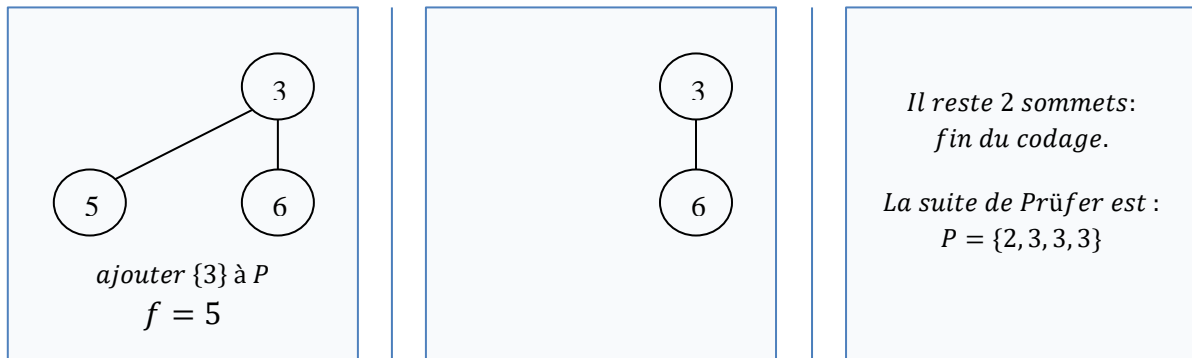
Ajouter à la suite de Prüfer P : le sommet s adjacent à f

Enlever de l'arbre G : le sommet f et l'arête incidente à f

Fin Tant que

Exemple 20 : Codage de Prüfer





Décodage :

Soit : $I = \{1, \dots, n\}$ un ensemble de n sommets.

Notons P une suite de Prüfer de $(n - 2)$ nombres tous appartenant à I .

Tant qu'il reste des éléments dans P et plus de 2 éléments dans I

Identifier le plus petit élément i de I n'apparaissant pas dans la suite P

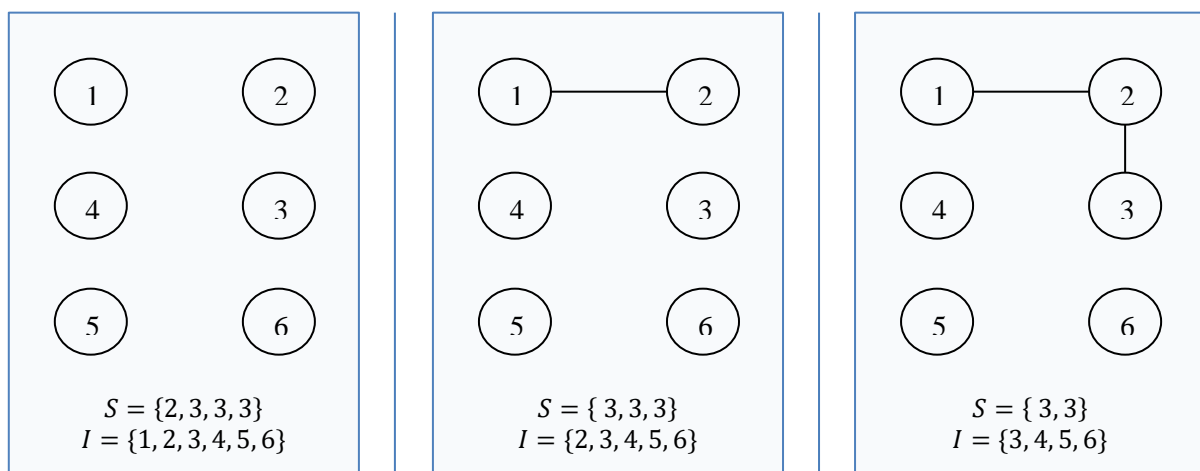
Relier par une arête le sommet i et le sommet s : premier élément de P

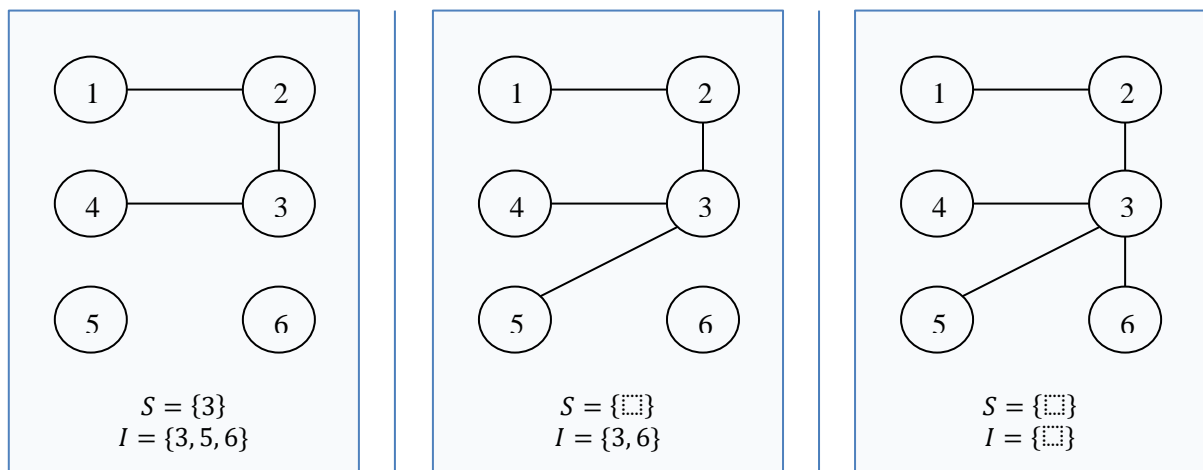
Enlever i de I et s de P

Fin Tant que

Ajouter l'arête reliant les deux sommets restants dans I .

Exemple 21 : Décodage de Prüfer





Théorème de Cayley :

Le nombre d'arbres que l'on peut construire sur n sommets numérotés ($n \geq 2$) est :

$$n^{n-2}$$

Généralisation :

Le nombre d'arbres couvrant un graphe complet K_n de degrés $d_1, d_2 \dots d_n$ est :

$$\frac{(n-2)!}{(d_1-1)! (d_2-1)! \dots (d_n-1)!}$$

Application 23 : Formule de Cayley

23.1. Vérifier la formule de Cayley pour $n = 3$ puis pour $n = 4$.
(Dessiner les arbres).

I.6.6.b. Arborescence

On appelle un arbre faisant intervenir l'orientation : une arborescence. (Faure, et al., 2014)

Dans une arborescence, nous distinguons :

La racine r : l'unique sommet qui n'a pas de prédécesseur.

A partir de r , il existe un chemin unique vers chacun des autres sommets.

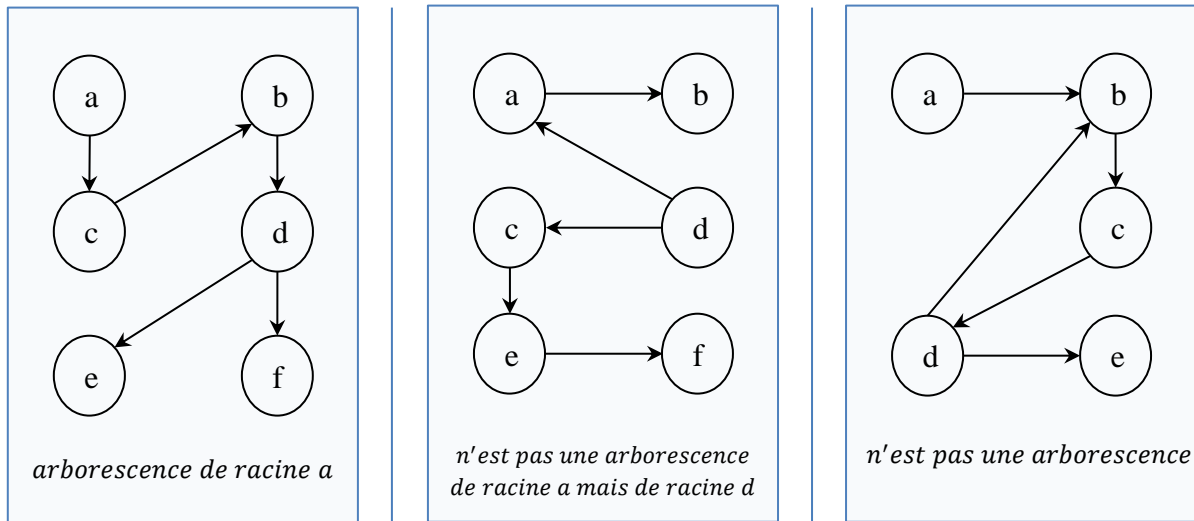
Les nœuds : les autres sommets de l'arborescence qui ont des successeurs.

Ils ont exactement 1 prédécesseur.

Les feuilles : les sommets qui n'ont pas de successeur.

Une branche : un chemin de la racine vers une feuille.

Exemple 22 : Arborescence



Les successeurs d'un sommet x sont appelés : descendants de x , et ses prédécesseurs : ascendants.

Lorsque chaque sommet a au plus 2 descendants, on parle d'arborescence binaire.

Application 24 : Voyage du professeur

24.1. Un professeur souhaite voyager d'une ville A vers une ville B :

Il existe 5 villes intermédiaires :

c : située à 30 km de A

d : située à 50 km de c

e : située à 70 km de A

f : située à 25 km de e et à 50 km de d

g : située à 75 km de c

La ville d'arrivée B est située à 35 km de f et à 30 km de g

Représenter les voyages possibles du professeur à l'aide d'un graphe.

I.6.7. Graphe eulérien

Cette notion s'applique aux chaînes, chemins, cycles et circuits.

On appelle chaîne eulérienne une chaîne passant par toutes les arêtes du graphe une fois et une seule.

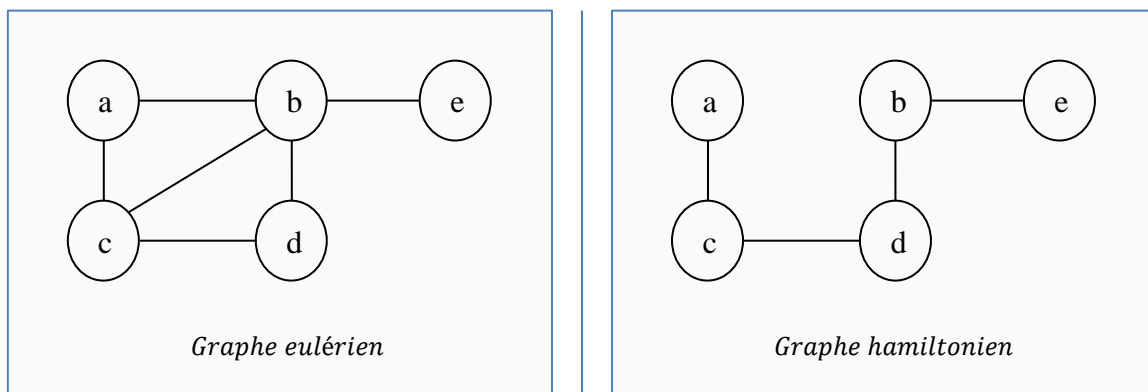
Un graphe est dit eulérien s'il possède un cycle eulérien ou une chaîne eulérienne.

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

Application 25 : Graphe eulérien

- 25.1. Est-il possible de dessiner un carré avec ses deux diagonales sans lever le crayon ?
- 25.2. Donner un critère permettant de savoir si un graphe est eulérien

Exemple 23 : Graphe eulérien et graphe hamiltonien



I.6.8. Graphe hamiltonien

Cette notion s'applique aux chaînes, chemins, cycles et circuits.

On appelle chaîne hamiltonienne une chaîne passant par tous les sommets du graphe une fois et une seule.

Un graphe possédant un sommet de degré 1 ne peut pas être hamiltonien.

Si un sommet dans un graphe est de degré 2, alors les deux arêtes incidentes à ce sommet doivent faire partie du cycle hamiltonien.

Les graphes complets K_n sont hamiltoniens.

Théorème d'Ore :

Soit G un graphe simple d'ordre $n > 3$.

Si pour toute paire de sommets non adjacents $\{x, y\}$ on a : $d(x) + d(y) > n$,
alors : G est hamiltonien.

Cas particulier de Dirac :

Soit G un graphe simple d'ordre $n > 3$.

Si pour tout sommet x de G , on a $d(x) > \frac{n}{2}$,

alors : G est hamiltonien.

Application 26 : Cas particulier de Dirac

26.1. Démontrer le cas particulier de Dirac.

I.7. Stabilité et absorption

I.7.1. Stabilité

Un ensemble stable est un ensemble de sommets non reliés 2 à 2.

Etant donné un graphe G , le cardinal maximal d'un ensemble stable de G est appelé nombre de stabilité de G . Il est noté α_G .

I.7.2. Absorption

Un ensemble absorbant est un ensemble de sommets tel que tout autre sommet du graphe est adjacent à au moins l'un d'entre eux.

Etant donné un graphe G , le cardinal minimal d'un ensemble absorbant de G est appelé nombre d'absorption de G . Il est noté β_G .

Un ensemble, à la fois stable et absorbant est appelé noyau.

Exemple 24 : Noyau d'un graphe

Un ensemble stable :

$\{a, e, h\}$

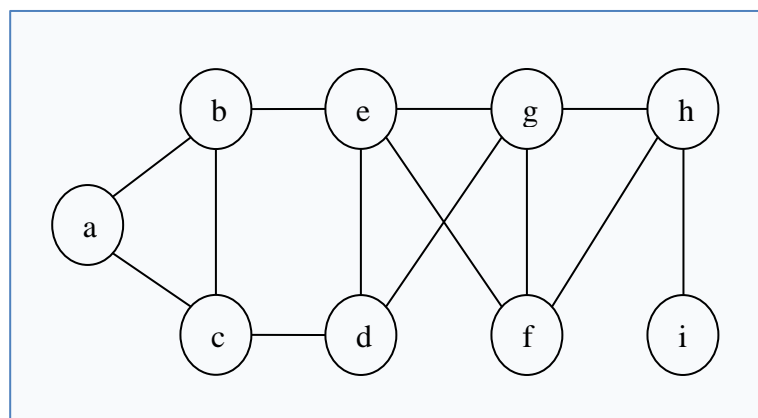
$$\alpha_G = 3$$

Un ensemble absorbant :

$\{a, e, h\}$

$$\beta_G = 3$$

$\{a, e, h\}$ est un noyau.



I.8. Pondération d'un graphe

Dans l'*Exemple 08* :

La relation (arc) qui lie les sommets (clubs) comporte une information (le nom du joueur).

Dans l'*Application 24* :

L'arc qui lie les sommets (ville) comporte une information (la distance).

Selon la nature du problème, les arcs (ou parfois les sommets) peuvent représenter une longueur, un coût, un poids, ...

Notons : $G = (X, U)$ un graphe,

On définit dans U la fonction de pondération :

$$\begin{aligned} I : U &\rightarrow \mathbb{R} \\ u &\rightarrow I(u) \end{aligned}$$

$I(u)$ est appelé poids de l'arc/arête u .

De même, on définit dans X la fonction de pondération :

$$\begin{aligned} V : X &\rightarrow \mathbb{R} \\ x &\rightarrow V(x) \end{aligned}$$

$V(x)$ est appelé poids du sommet x .

Le graphe G muni de l'une des deux fonctions de pondération est alors appelé graphe pondéré ou graphe valué.

Lorsque les valeurs affectées aux arcs/arêtes (ou aux sommets) sont des chaînes de caractères, ou du texte, on parle de graphe étiqueté.

Le poids d'une chaîne/ chemin est la somme des poids des arêtes/ arcs qui la/ le composent.

Jusqu'ici, la construction d'un graphe n'avait rien d'aléatoire, les composantes du graphe étaient fixes. Dans ce qui suit, nous allons introduire brièvement une certaine dose d'aléatoire.

I.8.1. Graphe probabiliste

Un graphe probabiliste est un graphe orienté pondéré dans lequel :

- il y a au plus un arc d'un sommet à l'autre,
- la somme des poids des arcs issus de chaque sommet est égale à 1.

On définit : la matrice carrée $M = (a_{i,j})$ d'ordre n telle que pour tout $(i, j) \in \llbracket 1, n \rrbracket^2$:

- si l'arc u allant du sommet i vers j existe, alors $a_{i,j} = I(u)$
- sinon, $a_{i,j} = 0$.

M est appelée matrice de transition associée au graphe probabiliste. Elle décrit le passage d'un état au suivant.

Un état probabiliste est une loi de probabilité sur l'ensemble des états possibles. Cette loi est représentée par une matrice ligne.

Matrice de transition :

Soit : M la matrice de transition d'un graphe probabiliste,

P_0 la matrice ligne décrivant l'état initial

et P_n l'état probabiliste à l'état n, ($n \in \mathbb{N}$).

Pour tout $n \in \mathbb{N}$: $P_n = P_0 \times M^n$

Convergence des états probabilistes:

Pour tout graphe probabiliste d'ordre 2 dont la matrice de transition M ne comporte pas de 0, la suite d'états $(P_n)_n$ converge vers un état P indépendant de l'état initial P_0

et P est l'unique solution de l'équation : $X = X.M$

où : $X = (x, (1 - x))$, $x \in [0,1]$.

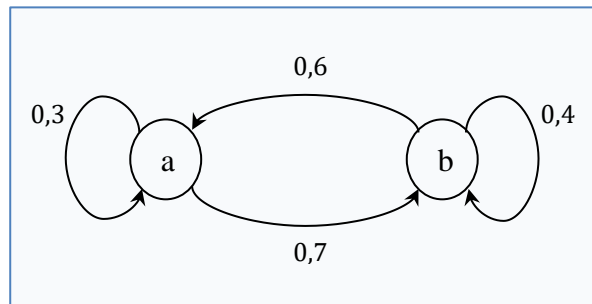
Exemple 25 : Graphe probabiliste

On considère la matrice de transition suivante :

$$M = \begin{pmatrix} 0,3 & 0,7 \\ 0,6 & 0,4 \end{pmatrix}$$

Soit l'état initial :

$$P_0 = (0,25 \quad 0,75)$$



Pour tout $n \in \mathbb{N}$:

$$M^n = \begin{pmatrix} \frac{6}{13} + \frac{7 \times (-3)^n}{13 \times 10^n} & \frac{7}{13} - \frac{7 \times (-3)^n}{13 \times 10^n} \\ \frac{6}{13} - \frac{6 \times (-3)^n}{13 \times 10^n} & \frac{7}{13} - \frac{6 \times (-3)^n}{13 \times 10^n} \end{pmatrix}$$

A l'état n : $P_n = P_0 \times M^n$

Après calcul : $\lim_{n \rightarrow +\infty} P_n = \left(\frac{6}{13} \quad \frac{7}{13} \right)$.

$P = \left(\frac{6}{13} \quad \frac{7}{13} \right)$ est donc l'état stable du graphe probabiliste.

II. Principaux algorithmes

Dans ce second chapitre, nous allons étudier un ensemble d'algorithmes sur les graphes. A chaque introduction d'un nouvel algorithme, le type du graphe sur lequel il est appliqué, le but de l'algorithme, le code de l'algorithme, ainsi qu'un ensemble d'applications sont proposés :

Problème de coloration

<i>II.1. Algorithme de coloration</i>	43
---	----

Problèmes de connexité

<i>II.2. Algorithme des CFC</i>	45
---------------------------------------	----

Arbres couvrants

<i>II.3. Algorithme de Kruskal</i>	46
<i>II.4. Algorithme de Prim</i>	48

Problèmes de chemins

<i>II.5. Algorithme de Bellman</i>	50
<i>II.6. Algorithme de Dijkstra</i>	53
<i>II.7. Algorithme général</i>	55
<i>II.8. Algorithme Ford</i>	57
<i>II.9. Algorithme de Ford-Fulkerson</i>	<i>Erreur ! Signet non défini.</i>
<i>II.10. Algorithme de Floyd-Warshall</i>	<i>Erreur ! Signet non défini.</i>

Problèmes d'ordonnancement

<i>II.11. Réseau PERT</i>	<i>Erreur ! Signet non défini.</i>
<i>II.12. Méthode MPM</i>	<i>Erreur ! Signet non défini.</i>

II.1. Algorithme de coloration

Type : Graphe non orienté

But : Colorer les sommets adjacents de couleurs différentes en utilisant un minimum de couleurs.

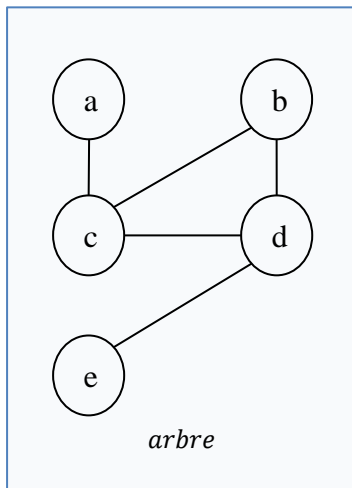
Une coloration avec k couleurs est donc une partition de l'ensemble des sommets en k stables.

Description :

Notons : $G = (X, U)$ un graphe,

On appelle nombre chromatique de G : le plus petit entier k pour lequel il existe une partition de X en k sous-ensembles stables. On le note $\gamma(G)$.

Exemple 26 : Algorithme de coloration



Dans l'*Exemple 26* :

Il nous faut au minimum trois couleurs pour colorer les sommets b , c et d (nommons ces couleurs respectives: 1, 2 et 3).

Une quatrième couleur n'est pas nécessaire puisque le sommet a peut prendre la même couleur que b ou d . De même, le sommet e peut prendre la même couleur que b ou c .

2 couleurs ne sont pas suffisantes puisque deux sommets adjacents parmi $\{b, c, d\}$ auraient la même couleur.

La coloration minimale n'est pas forcément unique.

Encadrement du nombre chromatique

Soit : G un graphe non orienté d'ordre n et de nombre de stabilité $\alpha(G)$.

Notons : $(m_i)_{1 \leq i \leq n}$ les degrés des sommets, $(G_j)_j$ les sous-graphes de G , et $(C_k)_k$ les cliques de G :

- 1) $\gamma(G) \leq \max_{1 \leq i \leq n} (m_i) + 1$
- 2) $\gamma(G) \leq n + 1 - \alpha(G)$
- 3) $\gamma(G) \geq \max_j \gamma(G_j)$
- 4) $\gamma(G) \geq \max_k |C_k|$

Application 27 : Nombre chromatique

- 27.1. Démontrer les encadrements précédents.
- 27.2. Encadrer le nombre chromatique de l'*Exemple 26*.

Algorithme de Welsh et Powell:

Etant donné un graphe non orienté : $G = (X, U)$,

Classer dans une liste : les sommets et leurs degrés dans un ordre décroissant.

Tant qu'il reste un sommet non coloré dans la liste

Attribuer une nouvelle couleur au sommet non coloré ayant le plus grand degré

Attribuer la même couleur aux sommets non adjacents à cette couleur

Fin Tant que

(Le nombre de couleurs trouvé par l'algorithme est supérieur ou égal au nombre chromatique).

Graphe parfait

Etant donné un graphe non orienté G .

Si pour tout sous-graphe induit G_0 de G : $\gamma(G_0) = |G_0|$

(le nombre chromatique est égal à sa plus grande clique)

alors le graphe G est appelé graphe parfait. (Berge, 1973)

Théorème des quatre couleurs

Les sommets d'un graphe planaire peuvent être colorés en utilisant au plus quatre couleurs de telle sorte que les sommets adjacents aient des couleurs différentes.

Applications :

Les applications de l'algorithme de coloration concernent en grande partie le domaine de la logistique, la coloration de cartes, le positionnement d'objets...

II.2. Algorithme des CFC

Type : Graphe orienté

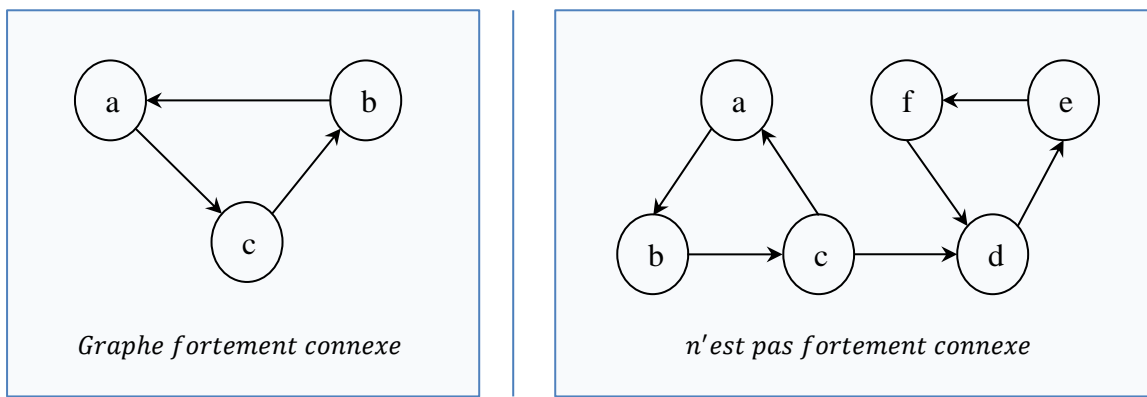
But : Regrouper les composantes fortement connexes (CFC) dans un graphe.

Description :

Notons : $G^o = (X, U)$ un graphe orienté,

G^o est fortement connexe si, entre tous sommets x et y quelconques, il existe un chemin c qui commence en x et arrive à y .

Exemple 27 : Graphe fortement connexe



Le graphe à droite n'est pas fortement connexe puisqu'il n'existe aucun chemin qui mène de d vers b . Par contre, il contient deux composantes fortement connexes CFC : $A = \{a, b, c\}$ et $B = \{d, e, f\}$.

Son graphe réduit est :



Algorithme :

Etant donné un graphe orienté : $G^o = (X, U)$,

Pour chaque sommet x de X .

Marquer du signe + :

x , les successeurs de x et les successeurs des sommets marqués de +

Marquer du signe - :

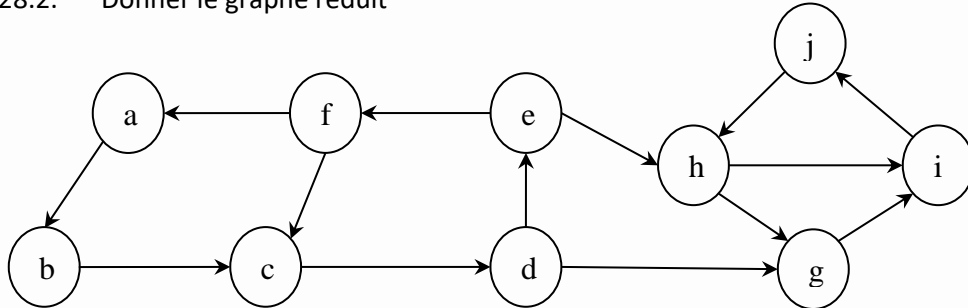
x , les prédécesseurs de x et les prédécesseurs des sommets marqués de -

La CFC est composée des sommets ayant les signes + et -

(effacer les signes + et - n'appartenant pas à la CFC et chercher une nouvelle CFC)

Application 28 : Composantes fortement connexes

- 28.1. Déterminer les composantes fortement connexes du graphe.
28.2. Donner le graphe réduit



Applications :

L'algorithme des composantes fortement connexes est fréquemment utilisé dans le domaine du transport et celui des réseaux.

II.3. Algorithme de Kruskal

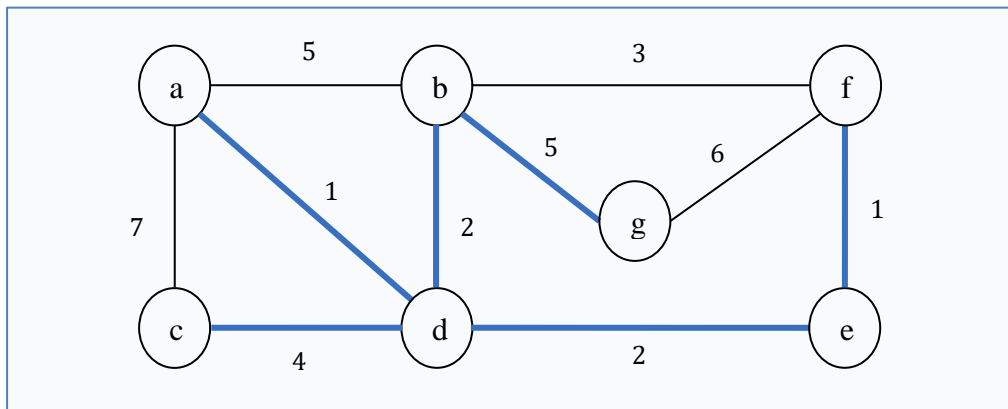
Type : Graphe non orienté pondéré

But : Dégager un arbre couvrant de poids minimal ou maximal

Description :

Notons : $G = (X, U)$ un graphe non orienté pondéré d'ordre n ,

Exemple 28 : Arbre couvrant minimal



Dans le graphe G précédent, l'arbre couvrant minimal (en bleu) est d'ordre 6, soit : $|G| - 1$.

Le poids de l'arbre est la somme des poids : $1 + 4 + 2 + 2 + 5 + 1 = 15$.

L'arête (b, f) ne figure pas dans l'arbre couvrant minimal malgré son petit poids, parce qu'elle formerait un cycle.

L'idée de former un arbre couvrant minimal consiste à emprunter les arêtes dans l'ordre croissant de leurs poids sans former de cycle. Et on s'arrête lorsqu'on aura emprunté $(n - 1)$ arêtes.

Algorithme :

Etant donné un graphe non orienté : $G = (X, U)$ d'ordre n ,
et I une fonction de pondération des arêtes.

Trier les arêtes de G dans un ordre croissant de leurs poids : $u_1, u_2 \dots u_m$.

$A = \emptyset$ (Initialisation de l'arbre couvrant minimal)

Pour i allant de 1 à m

 Si l'arête u_i ne forme pas un cycle dans A

 Ajouter u_i à A

 Fin Si

Fin Pour i

L'algorithme de Kruskal peut également être utilisé pour chercher un arbre de poids maximal, dans ce cas, les arêtes sont triées dans un ordre décroissant, et les plus grands poids sont pris en premier.

Application 29 : Algorithme de Kruskal

- 29.1. A partir de l'[Exemple 28](#), donner un arbre couvrant maximal.
- 29.2. Quel est le poids de cet arbre ?
- 29.3. L'arbre trouvé est-il unique ?

Applications :

L'algorithme de Kruskal est utilisé pour simplifier un câblage, supprimer les liaisons les moins rentables entre sites, etc.

II.4. Algorithme de Prim

Type : Graphe non orienté pondéré

But : Dégager un arbre couvrant de poids minimal ou maximal

Algorithme :

Initialisation:

Choisir un sommet de départ x , et le cocher par une *

La liste de sommets $P = \{x\}$

La liste des arêtes $A = \{\}$

Tant qu'il reste un sommet n 'appartenant pas à P faire

Cocher par une * les sommets adjacents à P

Choisir l'arête au plus petit poids ne formant pas de cycle

(parmi toutes les arêtes encadrées par 2 *)

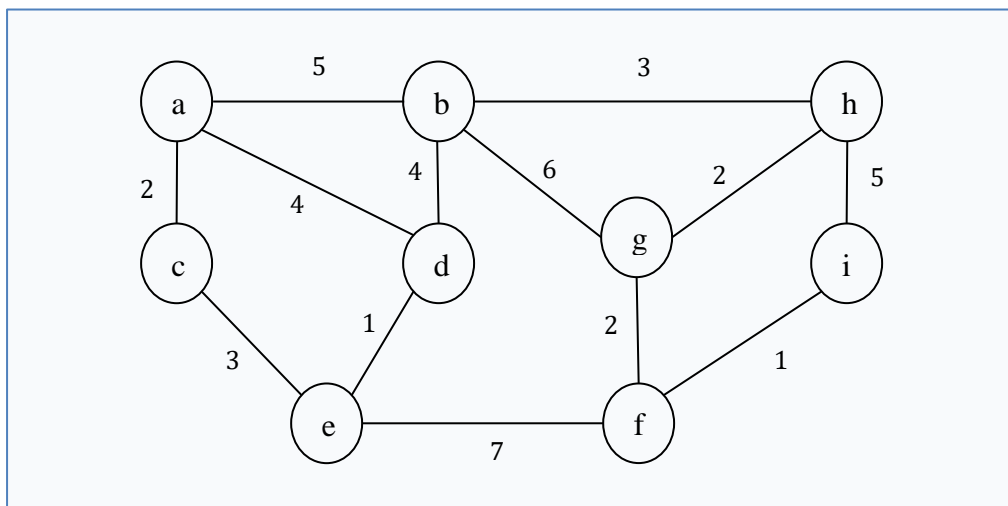
Ajouter le nouveau sommet à P et l'arête à A .

Fin tant que

Description :

Notons : $G = (X, U)$ un graphe non orienté pondéré d'ordre n ,

Exemple 29 : Algorithme de Prim



On initialise la liste des sommets par $P = \{a\}$

Il existe trois sommets adjacents : b, c et d .

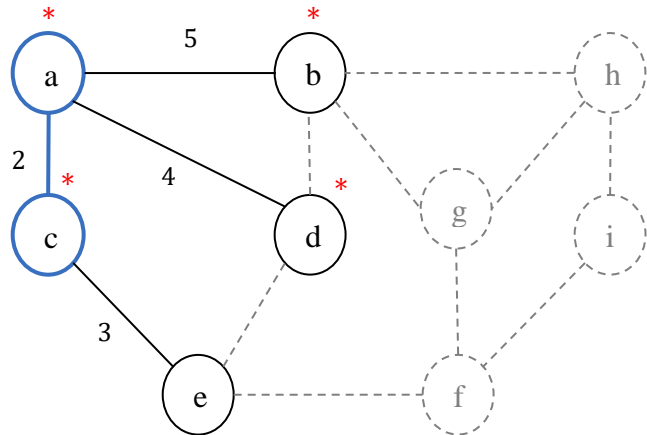
Parmi les arêtes : (a, b) , (a, c) et (a, d) , on choisit (a, c) qui a le plus petit poids : 2.

A cette itération :

En plus de (a, b) et (a, d), l'arête (c, e) est également considérée :

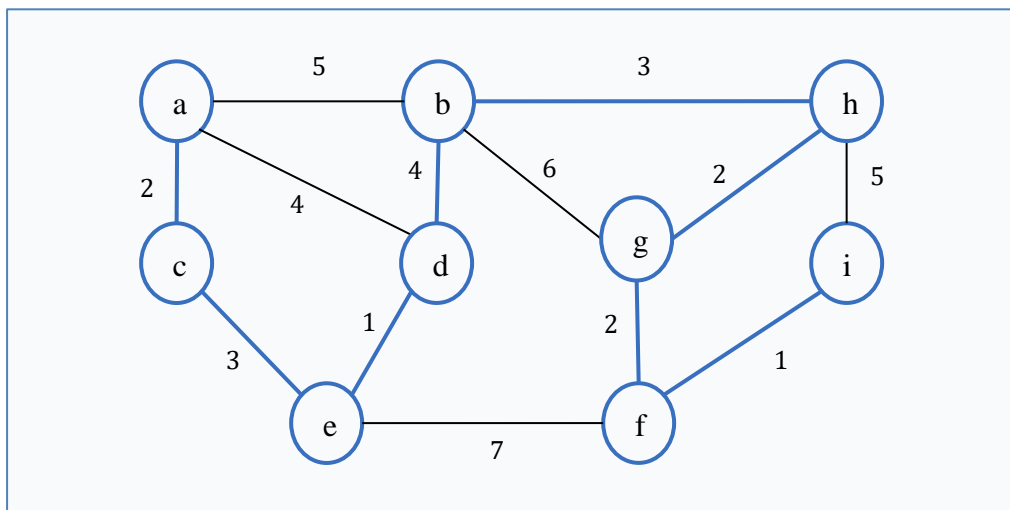
Le plus petit poids est 3 de (c, e).

Le nouveau sommet e ajoute deux nouvelles arêtes dans l'itération suivante : (e, d) et (e, f).



Le processus est réitéré jusqu'à ce que P contienne 9 sommets (taille de l'arbre = 8).

Finalement :



$$P = \{a, c, e, d, b, h, g, f, i\}$$

$$A = \{(a, c), (c, e), (e, d), (d, b), (b, h), (h, g), (g, f), (f, i)\}$$

Le poids de l'arbre couvrant minimal est : $2 + 3 + 1 + 4 + 3 + 2 + 2 + 1 = 18$.

Applications :

L'algorithme de Prim a les mêmes applications que l'algorithme de Kruskal : simplifier un câblage, supprimer les liaisons les moins rentables entre sites, etc.

Application 30 : Algorithme de Prim

- 30.1. A partir de l'*Exemple 29 : Algorithme de Prim*, donner un arbre couvrant maximal, en précisant son poids.

II.5. Algorithme de Bellman

Type : Graphe orienté pondéré (sans cycles)

But : Cet algorithme a deux utilisations possibles:

- Recherche du chemin minimal
- Recherche du chemin maximal

Algorithme :

Initialisation:

Initialiser le poids du sommet de départ par 0 ($m(s) = 0$)
Initialiser tous les autres par l'infini ($m(x) = \infty$ pour $x \neq s$)
Initialiser l'ensemble M de sommets par le point de départ ($M = \{s\}$)

Tant qu'il reste un sommet non traité faire

Prendre un sommet quelconque x ayant tous ses prédécesseurs dans M

Mettre à jour le poids de x

(choisir le plus court chemin qui mène à x) :

Si $\exists y \in X$ tel que $m(y) + I(y, x) < m(x)$

alors $m(x) \leftarrow m(y) + I(y, x)$

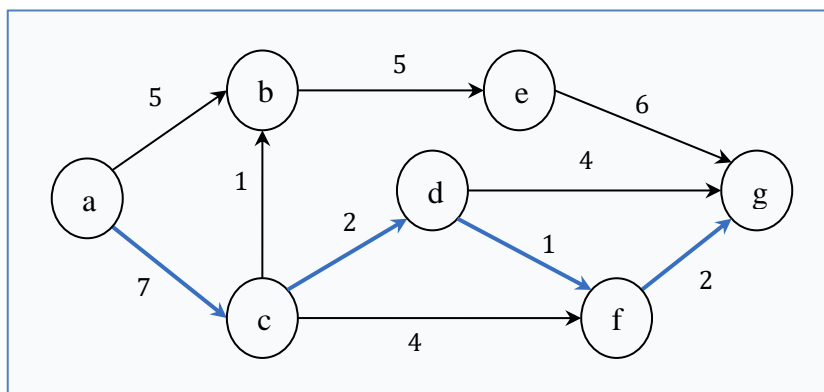
Ajouter x à M

Fin Tant que

Description :

Notons : $G^o = (X, U)$ un graphe orienté muni d'une fonction de pondération I sur U.

Exemple 30 : Algorithme de Bellman

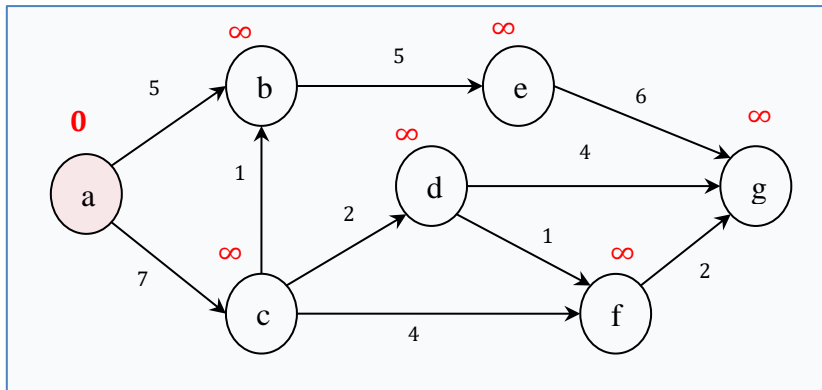


L'algorithme de Bellman passe donc par plusieurs itérations :

1. Initialisation :

Le sommet a prendra la valeur 0, et les autres sommets : ∞

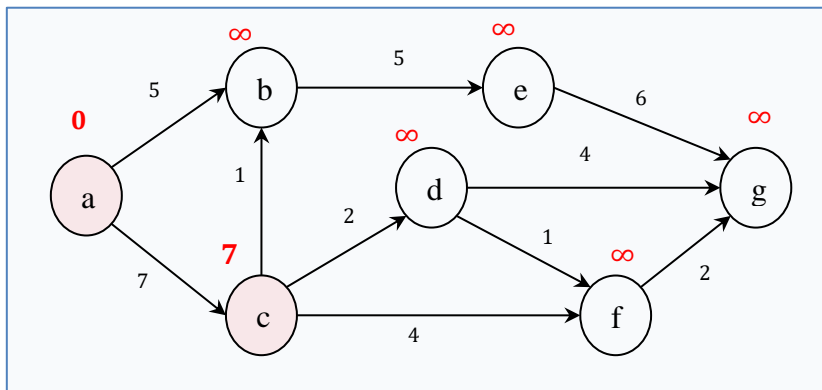
Seul le sommet a appartient à l'ensemble M



$$M = \{a\}$$

2. Itération 1 :

Seul le sommet c peut être considéré puisqu'il est le seul à avoir tous ses prédécesseurs dans M .



$$M = \{a, c\}$$

$$A = \{(a, c)\}$$

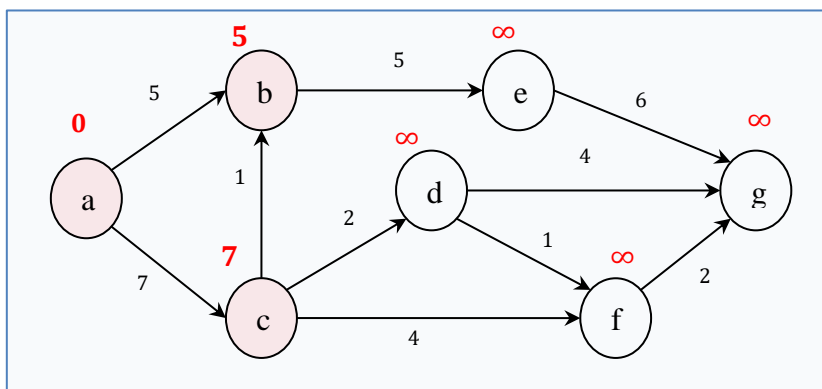
3. Itération 2 :

A la deuxième itération, nous avons le choix entre les sommets b et d . Prenons par exemple b : il y'a deux chemins possibles :

$\mu_1(a, b) = \{a, (a, b), b\}$ qui donne le poids 5,

et $\mu_2(a, b) = \{a, (a, c), c, (c, b), b\}$ qui donne le poids 8

C'est donc bien μ_1 qui sera retenu pour le poids de b .



$$M = \{a, c, b\}$$

$$A = \{(a, c), (a, b)\}$$

Nous poursuivons les itérations jusqu'à ce que qu'on parcoure tout le graphe.

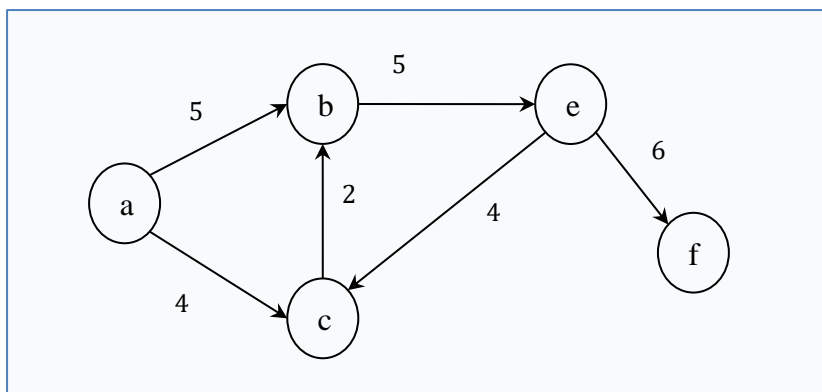
Finalement, pour déterminer le chemin optimal, nous commençons du dernier arc dans A, le sommet prédécesseur de g nous permet de remonter le graphe dans le sens inverse jusqu'à arriver au sommet de départ a.

Application 31 : Algorithme de Bellman

- 31.1. Dans l'exemple précédent, combien reste-t-il d'itérations ?
- 31.2. Terminer les itérations et donner la longueur du chemin.
- 31.3. Proposer un algorithme de Bellman qui retourne le plus long chemin.
- 31.4. Reprenons l'*Application 24 : Voyage du professeur*, utiliser l'algorithme de Bellman pour trouver le plus long chemin (empêcher le professeur de venir enseigner).

Limites :

Exemple 31 : Limites de l'algorithme de Bellman

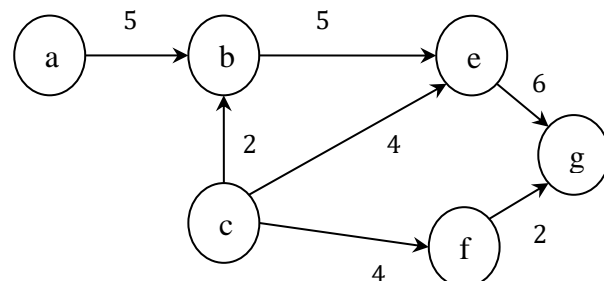


Dans cet exemple, qui a la particularité de contenir un cycle :

Après l'initialisation, nous avons : $M = \{a\}$. Aucun sommet n'a tous ses prédécesseurs dans M . Aucune itération n'est donc possible.

C'est aussi le cas du graphe :

(le sommet de départ a doit être unique)



Applications :

L'algorithme de Bellman est utilisé dans la recherche d'un chemin optimal : minimal (coût, temps, etc.) ou maximal (profit, couverture réseau, etc.) à condition qu'il n'y ait pas de cycle. Ces chemins optimaux peuvent aussi être exploités dans les problèmes d'ordonnancement.

II.6. Algorithme de Dijkstra

Type : Graphe orienté pondéré (sans valeurs négatives)

But : Trouver le chemin minimal

Algorithme :

L'algorithme de Dijkstra peut être appliqué aux graphes avec cycles

Initialisation:

Initialiser le poids du sommet de départ par 0 $(m(s) = 0)$
Initialiser tous les autres par l'infini $(m(x) = \infty \text{ pour } x \neq s)$

Tant que le sommet d'arrivée n'est pas atteint faire

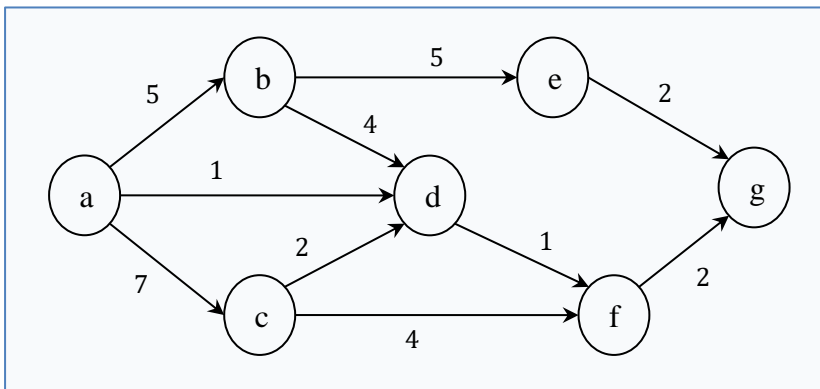
Choisir le sommet qui a le plus petit poids x
Traiter les arcs sortants de ce sommet

Fin tant que

Description :

Notons : $G^o = (X, U)$ un graphe orienté muni d'une fonction de pondération l sur U .

Exemple 32 : Algorithme de Dijkstra



L'initialisation donne au sommet de départ a la valeur 0 et aux autres sommets ∞ .

A la première itération, on choisit le sommet ayant le plus petit poids : a

Il existe 3 arcs sortants de a :

les sommets b , c , et d prennent donc les poids respectifs : 5, 7 et 1.

A l'itération suivante, le plus petit poids est celui de d ,

c'est donc le sommet d qui sera choisi, il a un seul arc sortant, on ne modifiera donc que le poids du sommet f .

A cette nouvelle itération, nous avons 3 sommets : b, c, et f. Celui au plus petit poids (f) sera considéré.

Les itérations se poursuivent jusqu'à atteindre le sommet d'arrivée g.

<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	$m(x)$
0	∞	∞	∞	∞	∞	∞	$a(0)$
	$5(a)$	$7(a)$	$1(a)$	∞	∞	∞	$d(1)$
	$5(a)$	$7(a)$		∞	$2(d)$	∞	$f(2)$
	$5(a)$	$7(a)$		∞		$4(f)$	$g(4)$

A partir de la dernière ligne, on remonte à la colonne du sommet g, elle correspond à 4(f). Le sommet f nous mène à d qui mène à son tour à a.

Le chemin optimal est donc $\mu * (a, g) = \{a, (a, d), d, (d, f), f, (f, g), g\}$ de poids : 4.

Interprétation :

A chaque itération, le sommet ayant le plus petit poids x est retenu pour l'itération suivante, et on n'y revient plus, puisque les autres sommets ont déjà des poids plus grands, s'il existe un arc de l'un de ces sommets y vers x, alors le nouveau poids est:

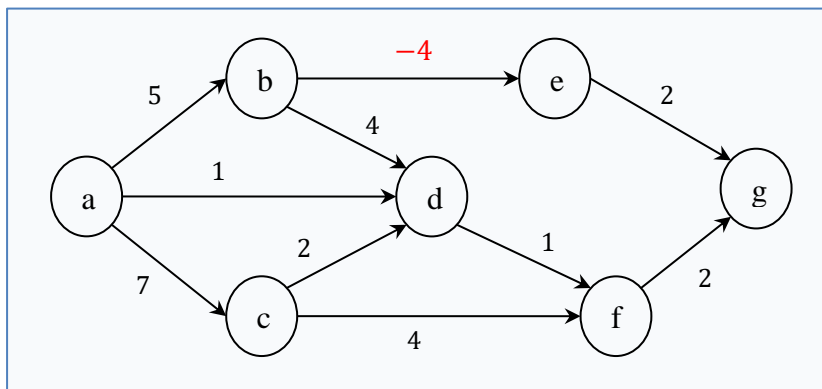
$$I(y, x) + m(y) > m(x) \quad (\text{car } m(y) > m(x) \quad \text{et} \quad I(y, x) > 0)$$

Limites :

L'interprétation précédente n'est plus valable si $I(y, x) < 0$.

Ainsi, l'algorithme de Dijkstra ne prend pas en considération les arcs qui ont une pondération négative comme le montre l'*Exemple 32 : Algorithme de Dijkstra* en

Exemple 33 : Limite de l'algorithme de Dijkstra



remplaçant $I(b, e)$ par -4 .

Le nouveau chemin optimal est : $\mu^*(a, g) = \{a, (a, b), b, (b, e), e, (e, g), g\}$ et son poids est : 3.

Applications :

L'algorithme de Dijkstra est utilisé dans le domaine de la logistique, dans les réseaux, dans l'ordonnancement des projets, et d'autres domaines ne faisant pas intervenir des valeurs négatives.

Application 32 : Algorithme de Dijkstra

32.1. En utilisant l'algorithme de Dijkstra, trouver le plus court chemin de
l'Application 24 : Voyage du professeur.

II.7. Algorithme général

Type : Graphe orienté pondéré (valeurs quelconques)

But : Trouver le chemin minimal/ arborescence minimale

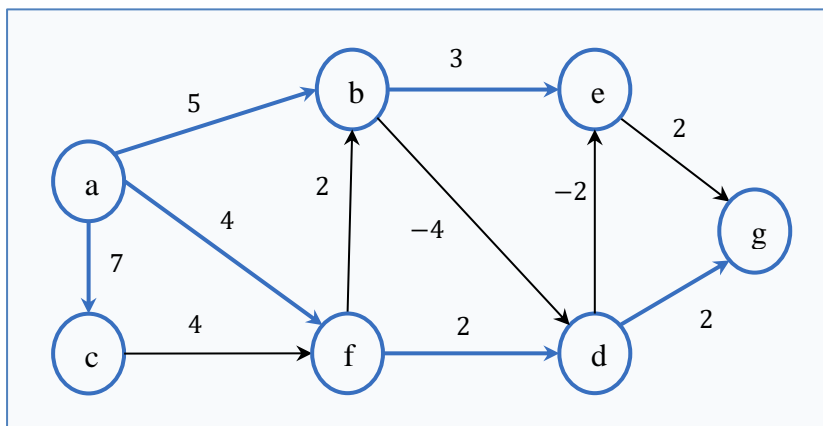
Description :

Notons : $G^o = (X, U)$ un graphe orienté muni d'une fonction de pondération I sur U .

On choisit dans un premier temps une arborescence :

(on peut par exemple utiliser l'algorithme de Dijkstra et le compléter)

Exemple 34 : Algorithme général



Il existe 5 arcs qui n'appartiennent pas à l'arborescence :

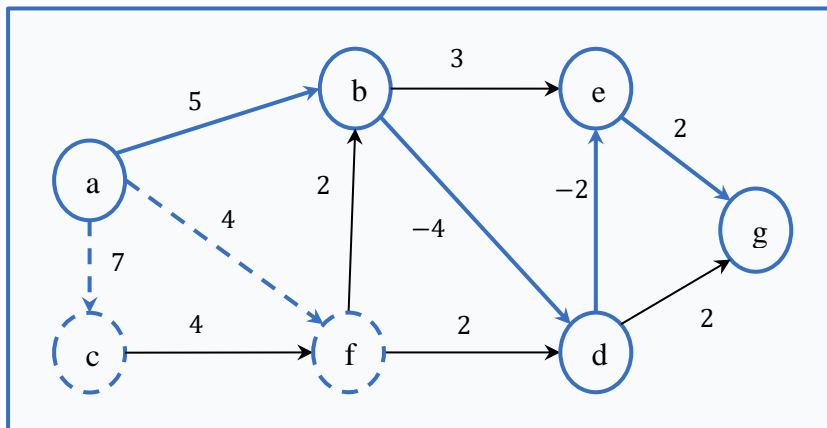
(c, f), (f, b), (b, d), (d, e) et (e, g).

- Considérons (f, b) : nous avons $I(f, b) + m(f) = 2 + 4 = 6$, $m(b) = 5$

Donc, il n'y aura pas de modification au niveau de l'arc (f, b).

- Considérons (b, d) : nous avons $I(b, d) + m(b) = -4 + 5 = 1$, $m(d) = 6$
Donc, l'arc (b, d) remplacera (f, d).
Le sommet d, et son successeur g prendront respectivement les poids : 1 et 3.

On réitère le processus pour les arcs n'appartenant pas à l'arborescence jusqu'à ce qu'il n'y ait plus aucune modification possible.



Finalement, le poids de l'arborescence optimale est 18.

Le chemin optimal est $\mu^*(a, g) = \{a, (a, b), b, (b, d), d, (d, e), e, (e, g)\}$, et son poids est :
 $5 - 4 - 2 + 2 = 1$.

Algorithme : (arborescence minimale)

Choisir une arborescence

Tant qu'une nouvelle itération est possible faire

Pour tout arc (x, y) n'appartenant pas à l'arborescence faire

Si $m(x) + I(x, y) < m(y)$ alors

Remplacer l'arc incident à y par (x, y)

Mettre à jour les poids des successeurs et leurs successeurs

Fin Si

Fin pour

Fin tant que

Déduire le chemin minimal à partir de l'arborescence minimale

Applications :

L'algorithme général est utilisé dans tous les domaines faisant intervenir les problèmes du chemin minimal ou d'arborescence minimale avec des valeurs quelconques.

Application 33 : Extension de l'algorithme général

- 33.1. Montrer que si l'arborescence minimale existe, alors le chemin minimal existe.
- 33.2. Est-il nécessaire d'utiliser l'algorithme général pour trouver le chemin maximal ? Justifier.
- 33.3. Appliquer l'algorithme général pour trouver le chemin minimal de l'*Exemple 33 : Limite de l'algorithme de Dijkstra*.

II.8. Algorithme de Ford

Type : Graphe orienté pondéré

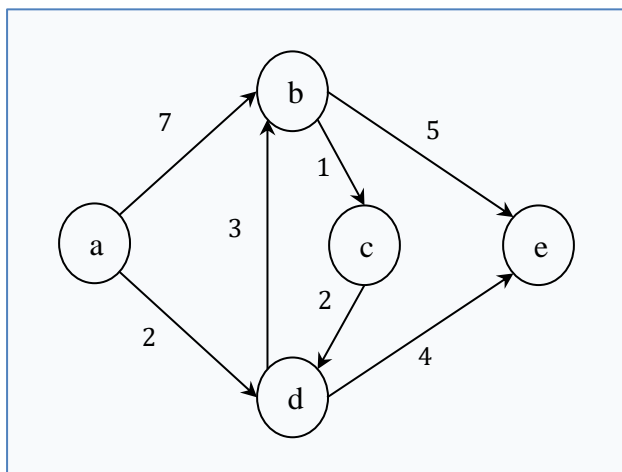
But : Trouver le chemin minimal

Description :

L'algorithme de Ford a le même principe que l'algorithme général.

Notons : $G^o = (X, U)$ un graphe orienté muni d'une fonction de pondération I sur U .

Exemple 35 : Algorithme de Ford



Nous avons : $X = \{a, b, c, d, e\}$, $U = \{(a, b), (a, d), (b, c), (c, d), (d, b), (b, e), (d, e)\}$

On initialise le sommet a par 0 et les autres sommets par ∞ .

A chaque itération:

Pour tout arc $(x, y) \in U$:

Si $m(x) + I(x, y) < m(y)$ alors $m(y) \leftarrow m(x) + I(x, y)$

Par exemple :

A la première itération, nous avons : $m(b) = \infty$, $m(a) + I(a, b) = 0 + 7$,

Donc le sommet b prendra la nouvelle valeur : 7 .

On s'arrête quand on obtient les mêmes valeurs dans 2 itérations successives, ou après $(|G^o| - 1) = 4$ itérations.

		$m(a)$	$m(b)$	$m(c)$	$m(d)$	$m(e)$
	Initialisation	0	∞	∞	∞	∞
Itération 1	(a, b)	0	7	∞	∞	∞
	(a, d)	0	7	∞	2	∞
	(b, c)	0	7	8	2	∞
	(c, d)	0	7	8	2	∞
	(d, b)	0	5	8	2	∞
	(b, e)	0	5	8	2	10
	(d, e)	0	5	8	2	6
Itération 2	(a, b)	0	5	8	2	6
	(a, d)	0	5	8	2	6
	(b, c)	0	5	6	2	6
	(c, d)	0	5	6	2	6
	(d, b)	0	5	6	2	6
	(b, e)	0	5	6	2	6
	(d, e)	0	5	6	2	6
Itération 3	(a, b)	0	5	6	2	6
	(a, d)	0	5	6	2	6
	(b, c)	0	5	6	2	6
	(c, d)	0	5	6	2	6
	(d, b)	0	5	6	2	6
	(b, e)	0	5	6	2	6
	(d, e)	0	5	6	2	6

Algorithme :

Initialisation:

Initialiser le poids du sommet de départ par 0 $(m(s) = 0)$
 Initialiser tous les autres par l'infini $(m(x) = \infty \text{ pour } x \neq s)$

Répéter jusqu'à stationnarité (maximum : $(|G^o| - 1)$ fois)

Pour tout arc $(x, y) \in U$:

Si $m(x) + I(x, y) < m(y)$
 alors $m(y) \leftarrow m(x) + I(x, y)$

Fin Si

Fin Pour

Applications :

L'algorithme de Ford est utilisé pour trouver un chemin minimal. Il peut également être utilisé pour trouver le chemin maximal.

Application 34 : Algorithme de Ford

- 34.1. Proposer un algorithme de Ford cherchant le chemin maximal.
- 34.2. Changer l'ordre des arcs dans l'exemple de telle sorte à avoir stationnarité après une seule itération.
- 34.3. Utiliser l'algorithme de Ford pour trouver le chemin minimal du graphe suivant :

