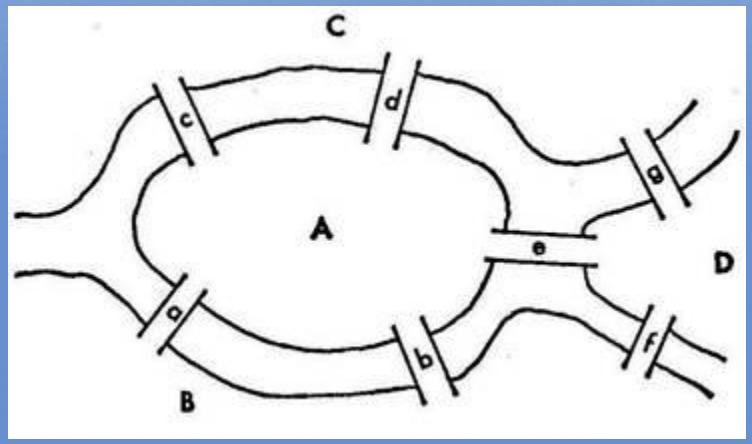
Théorie des graphes



Licence 3 Génie Logiciel

Année Académique: 2024-2025

ECOLE DES MINES, DE L'INDUSTRIE ET DE LA GEOLOGIE

Alio Boubacar Goga, PhD Researcher

Plan du cours

Plan du cours

Section 1 : Introduction

Section 2 : Les notions de base sur la théorie des graphes

Section 3 : Arbres et Arborescences

Section 4 : Le parcours de graphe

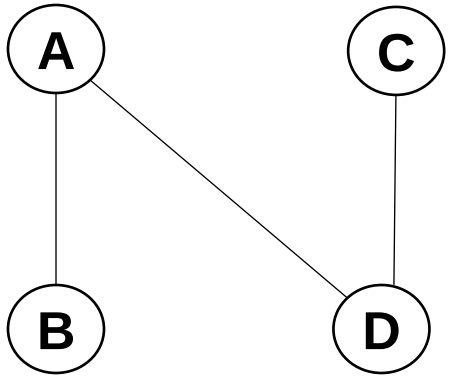
Organisation du cours

- **C**ours **M**agistral (**CM**) : : 16 h
- **T**ravaux **D**irigés (**TD**) : 10 h

gogaalioboubacar@gmail.com

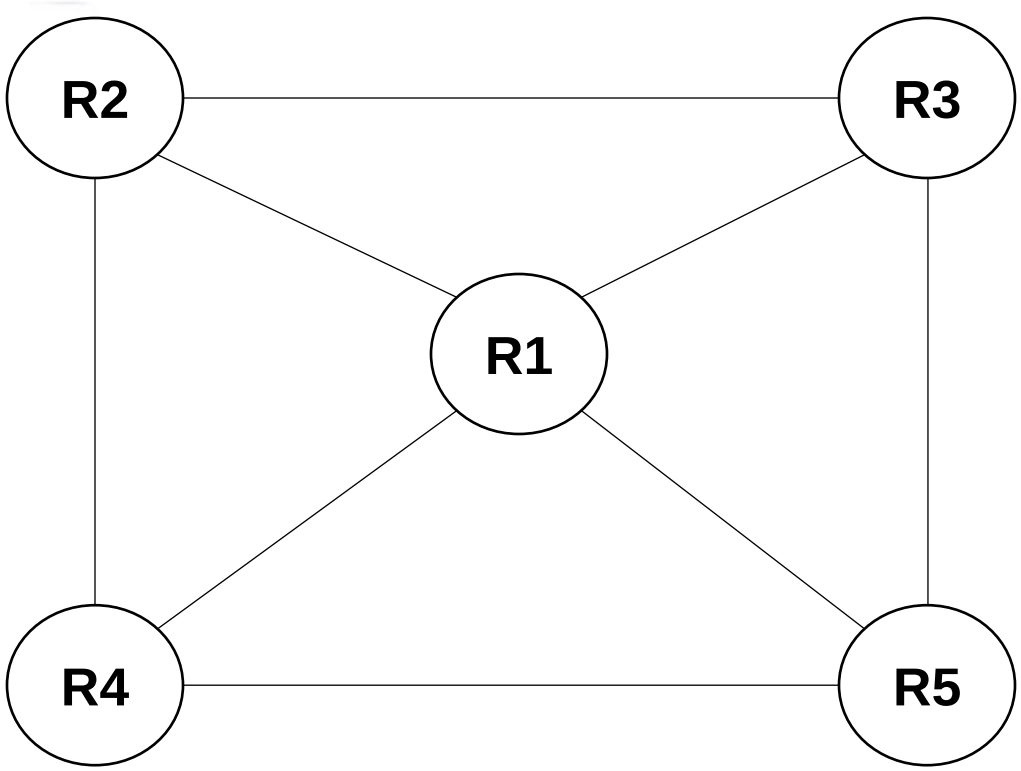
Section 1 : Introduction

- La théorie des graphes est une discipline mathématiques et informatique.
- Elle étudie les graphes.
- Les graphes modélisent de nombreuses situations concrètes où interviennent des objets en interaction.
- Un graphe est un ensemble de nœuds ou sommets (qui peuvent représenter par exemple des villes) liés par des arcs, qui seraient alors des routes.
- Représenter un graphe, c'est utiliser :
- des cercles (équivalent aux sommets)
- des lignes (équivalent aux arcs)



- Le graphe ci-dessous représente des ronds-points et les liens qui existent entre ces ronds-points.

Section 1 : Introduction



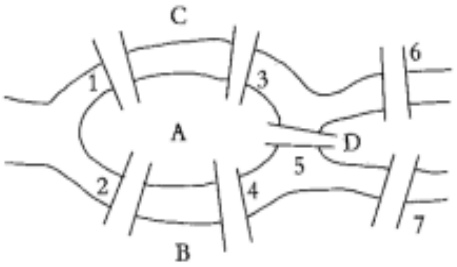
- Les ronds-points sont donc les nœuds. L'arc allant de R2 à R3 indique la présence d'un lien direct entre ces deux ronds-points. Il est noté (R2, R3) ou (R3, R2) selon le sens voulu.

Section 1 : Introduction

- Par contre pour aller de R3 à R4, il n'y a pas de lien direct, il faudra par exemple passer par R1 si on ne souhaite qu'un changement.
- Un **chemin** permet de rejoindre différents sommets liés entre eux par des arcs. Par exemple **R2- R1- R5**, est un chemin de longueur 2 (la longueur est le nombre d'arcs suivis) .
- On parle de **circuit** lorsqu'on peut partir d'un nœud et y revenir. Ici, le graphe contient de nombreux circuits, comme **R2- R1- R4- R2**.
- L'**ordre** d'un graphe correspond au nombre de sommets qu'il contient. Notre exemple contient 5 ronds-points. Donc, il s'agit d'un graphe d'ordre **5**.
- Deux nœuds sont dits adjacents (ou voisins) s'il existe un lien permettant d'aller de l'un à l'autre . Donc **R4** est adjacent à **R1**, **R2** et **R5**.

Section 1 : Introduction

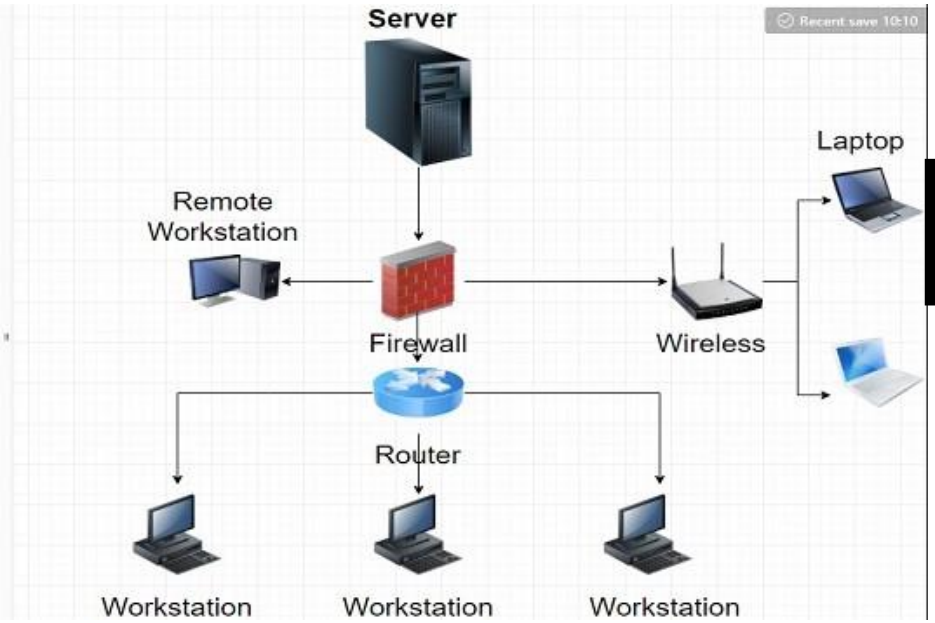
- Le problème des sept (7) ponts de Königsberg: la théorie des graphes est un
- outil puissant de modélisation et de résolution de problèmes concrets. A
- l'origine, elle était présentée comme une curiosité mathématique ;
- Euler lors d'une de ses promenades nocturnes a voulu tracer un itinéraire
- circulaire dans la ville de Königsberg.
  - Partant d'un point donné, il voulut visiter les sept ponts de cette ville une seule fois seulement, puis retourner à son point de départ. La figure ci-dessous montre les 7 ponts de Königsberg.

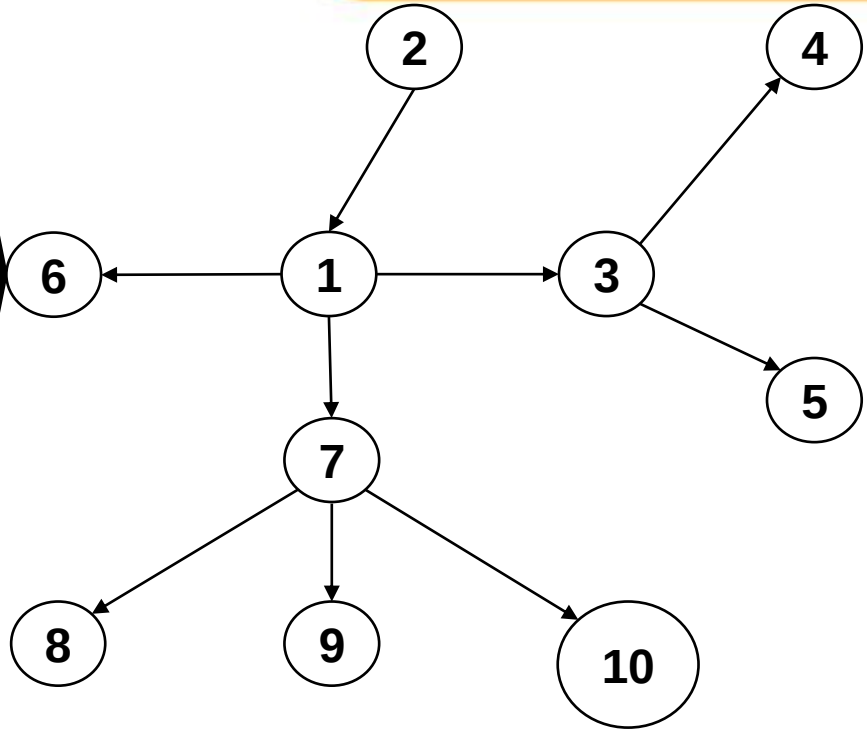


- Notons que les points A, B, C et D sont des rives
- Question : cela est-il possible ?

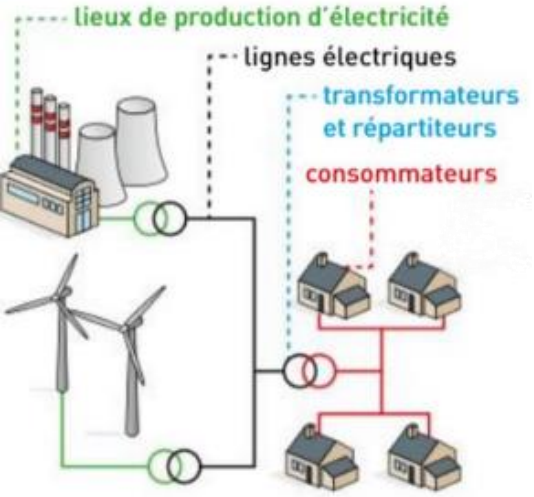
Section 1 : Introduction

Exemple de graphe: Réseau informatique





Modélisation du réseau informatique

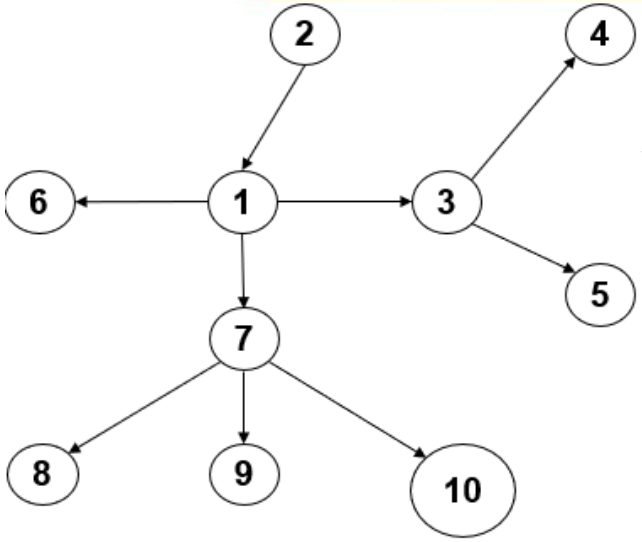


Question: Modéliser ce **Réseau électrique**.

Section 2 : Les notions de base sur la théorie des graphes

- Les graphes sont des concepts mathématiques utilisés pour modéliser des
- relations binaires entre des objets d'un même ensemble. Ils sont
- fréquemment utilisés pour modéliser des systèmes qui se présentent sous la
- forme d'un réseau. Il existe deux types de graphes :
  - les graphes orientés et
  - les graphes non orientés.
- un graphe orienté est un couple (X, U), où X est l'ensemble des sommets du graphe et U est l'ensemble de ses arcs. X et U sont finis.
  - L'arc est une relation entre deux sommets, doté d'une orientation : si u = (x, y) est un arc de U, avec x, y ∈ X, la relation est orientée de x vers y.
  - le graphe G est noté G = (X, U).

L'exemple ci-dessous montre un graphe orienté:

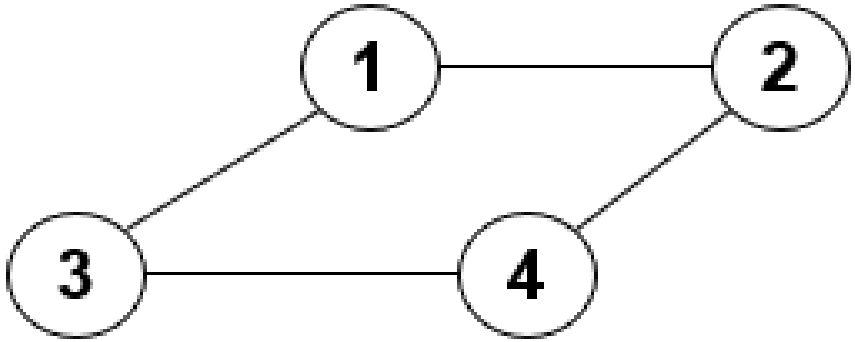


A retenir :

- ✓ une boucle est un arc reliant un sommet à lui-même.
- ✓ un graphe orienté est dit élémentaire s'il ne contient pas de boucle

- ❑ un graphe non orienté est défini par le couple (X, E), où X est l'ensemble des sommets du graphe et E, l'ensemble de ses arêtes. Si x, y sont en relation, cette dernière est décrite par l'arête e=(x, y).
- ○ dans ce type de graphe, le sens de la relation n'est pas invoqué.

L'exemple ci-dessous montre un graphe non orienté:



A retenir :

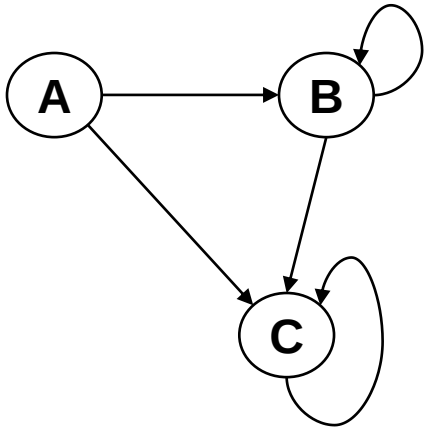
- ✓ une boucle est une arête reliant un sommet à lui-même.
- ✓ Un graphe non-orienté est dit simple s'il ne comporte pas de boucle, et s'il ne comporte jamais plus d'une arête entre deux sommets.
- ✓ On appelle ordre d'un graphe le nombre de ses sommets, c'est-à-dire card(X).
- ✓ On appelle taille d'un graphe le nombre de ses arêtes c'est-à-dire card(E).

❖ Représentation d'un graphe par une matrice d'adjacence:

- En pratique, la représentation visuelle n'est pas commode. Donc, on utilise la représentation matricielle.
- La matrice d'adjacence d'un graphe simple est une matrice N×N, ses éléments prennent deux valeurs 1 ou 0. Chaque ligne et chaque colonne correspondent à un sommet du graphe.
- Ainsi chaque élément de la matrice indique la relation qui existe entre deux sommets :
  - 1 signifie que les deux sommets sont reliés par un arc orienté.
  - 0 signifie que les deux sommets ne sont pas reliés par un arc orienté.

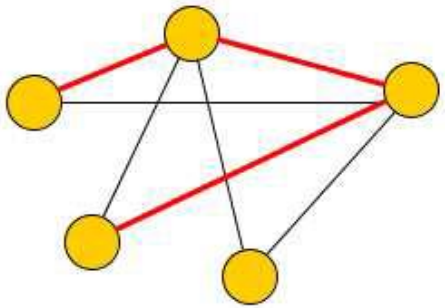
Section 2 : Les notions de base sur la théorie des graphes

- Donner la matrice d'adjacence de ce graphe ci-dessous:

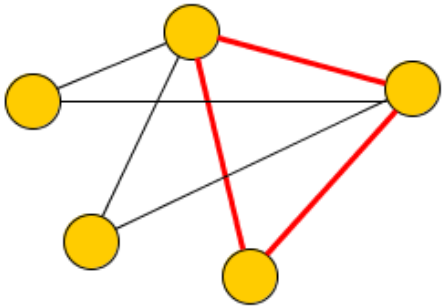


$$\begin{array}{c} \nearrow \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{ccc} 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \end{array} \quad \begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

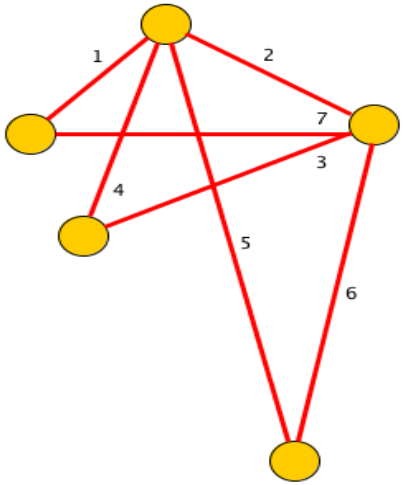
- Un chemin est une suite de nœuds v1, v2, ..., vn telle que pour tout 1 ≤ i ≤ n-1, les nœuds vi, vi+1 sont reliés par une arête:



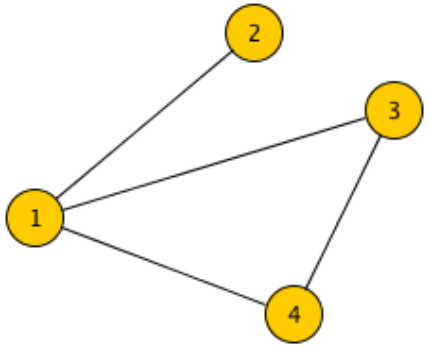
- Un cycle est un chemin v1, v2, ..., vn tel que v1 = vn:



- Un cycle eulérien est un cycle qui passe exactement une fois par chaque arête du graphe



- Un graphe eulérien est un graphe qui contient un cycle eulérien.
- Ce graphe est-il eulérien ? Non



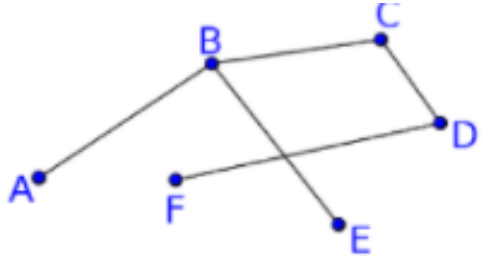
Le degré d'un nœud X, noté deg(X) est le nombre d'arêtes ayant ce nœud pour extrémité.

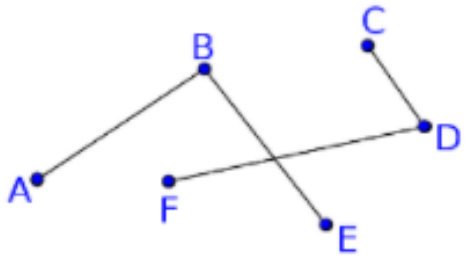
deg(1) = 3; deg(2) = 1; deg(3) = 2; deg(4) = 2

Théorème (Euler, 1736)

Un graphe connexe G est eulérien si et seulement si chacun de ses nœuds a un degré pair.

- Un graphe est dit connexe si deux de ses sommets quelconques sont toujours reliés par une chaîne. Ces deux (2) graphes sont-ils connexes ?



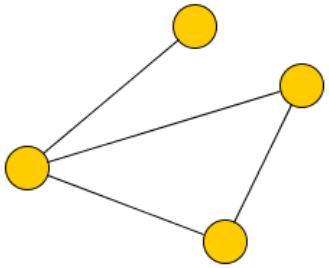


Théorème

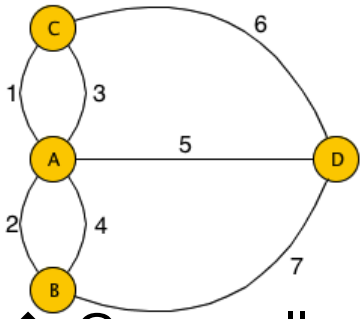
Un graphe connexe possède un parcours eulérien si et seulement si, il possède au plus deux nœuds de degré impair.

Section 2 : Les notions de base sur la théorie des graphes

- Un parcours eulérien est un chemin qui passe exactement une fois par chacune des arêtes d'un graphe.



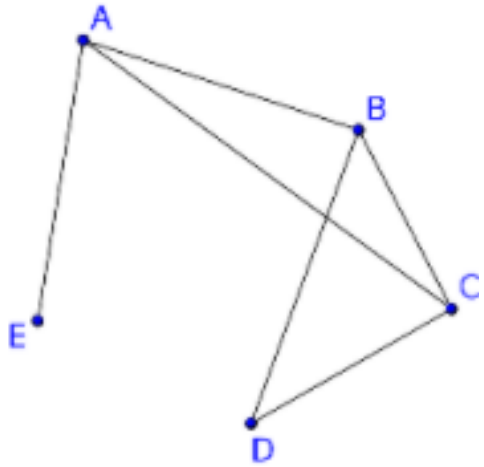
Ce graphe possède un parcours eulérien. Cependant, pas de cycle eulérien.



ce graphe n'est pas eulérien, car il possède des nœuds de degré impair.

- On appelle chaîne toute succession d'arêtes dont l'extrémité de l'une (sauf la dernière) est l'origine de la suivante. Le nombre d'arêtes qui composent une chaîne est appelé longueur de la chaîne.
- on appelle chaîne fermée toute chaîne dont l'origine et l'extrémité coïncident.

❖ Exemple : chaines et cycles



E-A-C-B est une chaîne de longueur 3.

E-A-C-B-A-E est une chaîne fermée de longueur 5. Ce n'est pas un cycle car l'arête A-E est parcourue deux fois.

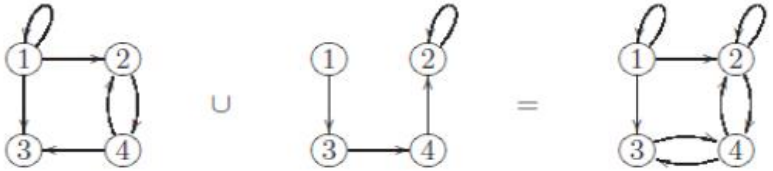
D-B-A-C-D est un cycle de longueur 4.

Remarque : Un graphe est complet lorsque chacun de ses sommets est relié à tous les autres. Un graphe complet est donc nécessairement connexe mais la réciproque est fausse comme le montre l'exemple ci-dessus.

❖ Le nombre de sommet du graphe est appelé l'ordre d'un graphe.

- L'union de 2 graphes: **G1UG2 = <S,A1>U<S,A2> = <S,A1UA2>**

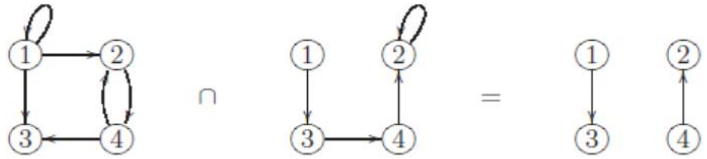
- **M1UM2** est définie par **(M1UM2)[i,j]=M1[i,j]ᵛM2[i,j]:**



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

- L'intersection de 2 graphes: **G1∩G2 = <S,A1> ∩ <S,A2> = <S,A1 ∩ A2>**

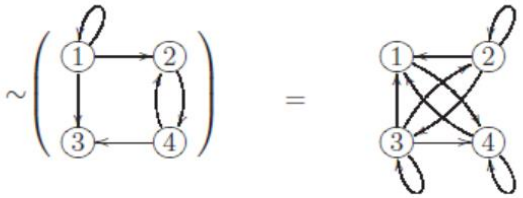
- **M1 ∩ M2** est définie par **(M1 ∩ M2)[i,j]=M1[i,j]ᴧM2[i,j]:**



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

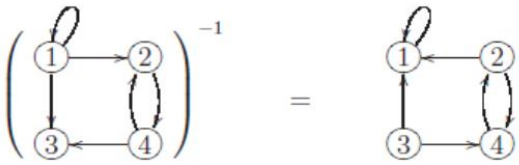
Section 2 : Les notions de base sur la théorie des graphes

- Le complément d'un graphe : le complément d'une matrice est obtenu en faisant la négation des entrées
- **~G= ~ <S, A> = <S, ~ A>. ~ M est défini par (~M)[i,j]=¬(M[i,j]).**



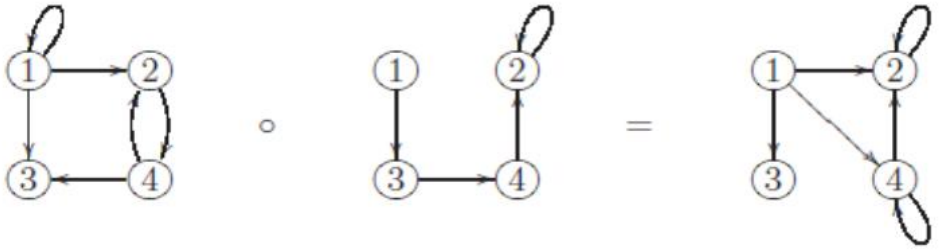
$$\sim\begin{pmatrix}1&1&1&0\\0&0&0&1\\0&0&0&0\\0&1&1&0\end{pmatrix}=\begin{pmatrix}0&0&0&1\\1&1&1&0\\1&1&1&1\\1&0&0&1\end{pmatrix}$$

- G^{-1}= <S,A>$^{-1}$= <S,A^{-1}>
- **M^{-1} est défini par (M^{-1}) [i,j]=M[j,i].**



$$\begin{pmatrix}1&1&1&0\\0&0&0&1\\0&0&0&0\\0&1&1&0\end{pmatrix}^{-1}=\begin{pmatrix}1&0&0&0\\1&0&0&1\\1&0&0&1\\0&1&0&0\end{pmatrix}$$

- Produit de 2 graphes: le produit des matrices est obtenu en faisant une opération similaire au produit matrice en algèbre linéaire, en remplaçant + par **∨** et • par **∧.**



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

- La définition de la somme booléenne et du produit booléen est:

$$0 \oplus 0 = 0 \qquad 0 \otimes 0 = 0$$
$$1 \oplus 0 = 1 \qquad 1 \otimes 0 = 0$$
$$0 \oplus 1 = 1 \qquad 0 \otimes 1 = 0$$
$$1 \oplus 1 = 1 \qquad 1 \otimes 1 = 1$$

Propriétés des relations sous forme matricielle:

- Totalité: au moins un 1 par ligne
- Déterminisme: au plus un 1 par ligne
- Surjectivité: au moins un 1 par colonne
- Injectivité: au plus un 1 par colonne
- Application: exactement un 1 par ligne
- Application bijective: exactement un 1 par ligne et par colonne

	$\begin{pmatrix}1&0&0\\0&0&1\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&0&0\\1&0&0\end{pmatrix}$	$\begin{pmatrix}0&0&0\\0&1&1\\1&0&1\end{pmatrix}$	$\begin{pmatrix}1&0&0\\0&1&0\\1&0&0\end{pmatrix}$	$\begin{pmatrix}0&1&0\\0&0&1\\1&0&0\end{pmatrix}$
Totalité	×			×	×
Déterminisme		×		×	×
Surjectivité			×		×
Injectivité					×
Application				×	×
Application bijective					×

❖ Fermeture transitive d'un graphe orienté non transitif:

➢ **Propriété 1** : G est transitif si ∀ u,v,w ∈ X, u≠v≠w

$$\begin{cases} (u,v) \in E \\ (v,w) \in E \end{cases} \quad \text{donc } (u,w) \in E$$

❑ Si G n'est pas transitif, on le complète pour obtenir sa fermeture transitive de manière à satisfaire la **Propriété 1** ci-dessus.

❑ L'algorithme suivant construit la fermeture transitive d'un graphe :

(1) On détermine la matrice d'adjacence

(2) On pose N=M k=2

(3) On calcul M^k

(4) Tester si N = N ⊕ M^k

Si oui, terminer N est la fermeture transitive

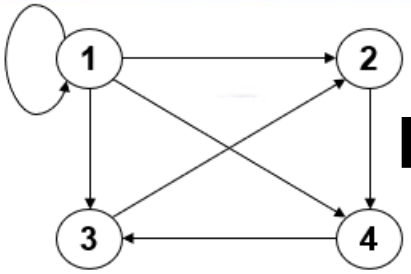
Sinon, aller en (5)

(5) On pose N = N ⊕ M^k

K=k+1

Aller en (3)

❖ Déterminer la fermeture transitive de ce graphe ci-dessous:



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

On pose N=M

K=2

$$\mathbf{N}=\begin{pmatrix}1&1&1&1\\0&0&0&1\\0&1&0&0\\0&0&1&0\end{pmatrix}$$

$$M^2=\begin{pmatrix}1&1&1&1\\0&0&1&0\\0&0&0&1\\0&1&0&0\end{pmatrix}$$

Tester si N = N $\oplus$ M^2 ; N $\oplus$ M^2 = $\begin{pmatrix}1&1&1&1\\0&0&1&1\\0&1&0&1\\0&1&1&0\end{pmatrix}$ **N $\oplus$ M^2 ≠N**

N=N $\oplus$ M^2 : Nouvelle valeur de N

Allons en (**5**) . Posons N=N $\oplus$ M^2 ; k=k+1 alors K=3 et allons en (**3**). Calculons M^3 .

$$M^3=\begin{pmatrix}1&1&1&1\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}$$

Tester si N = N $\oplus$ M^3 ; $N \oplus M^3=\begin{pmatrix}1&1&1&1\\0&1&1&1\\0&1&1&1\\0&1&1&1\end{pmatrix}$ **N $\oplus$ M^3 ≠N or N=N $\oplus$ M^2**

Plus précisément N $\oplus$ M^2 $\oplus$ M^3 ≠ N $\oplus$ M^2 ; **N= $N \oplus M^3$ =N $\oplus$ M^2 $\oplus$ M^3 : Nouvelle valeur de N**

Allons en (**5**) . Posons **N= N** $\oplus M^2 \oplus M^3$; k=k+1 alors K=4 et allons en (**3**). Calculons M^4 .

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

Tester si N = N $\oplus M^4$; $N \oplus M^4=\begin{pmatrix}1&1&1&1\\0&1&1&1\\0&1&1&1\\0&1&1&1\end{pmatrix}+\begin{pmatrix}1&1&1&1\\0&0&0&1\\0&1&0&0\\0&0&1&0\end{pmatrix}$

$N \oplus M^4=\begin{pmatrix}1&1&1&1\\0&1&1&1\\0&1&1&1\\0&1&1&1\end{pmatrix}$=**N or N = N** $\oplus M^2 \oplus M^3$ **:** C'est l'ancienne valeur de N

donc $N \oplus M^4$ (nouvelle valeur de N) est égale à l'ancienne valeur de N

$N \oplus M^4 =$**N** $\oplus M^2 \oplus M^3 \oplus M^4=$**N:** c'est la nouvelle valeur de N

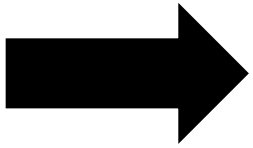
N = N $\oplus M^4$ **donc N** $\oplus M^2 \oplus M^3$ **= N** $\oplus M^2 \oplus M^3 \oplus M^4$

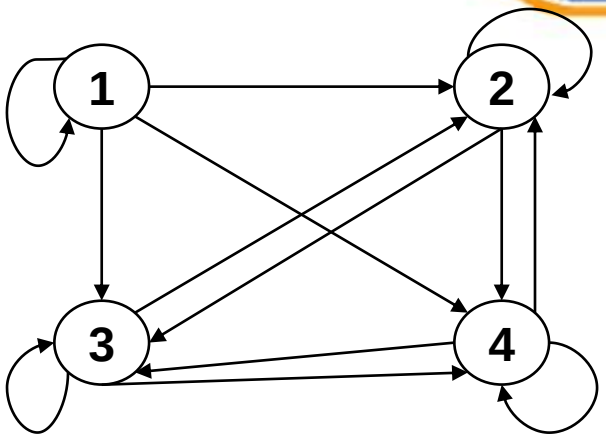
C'est l'ancienne valeur de N

Oui, terminer la nouvelle N est la fermeture transitive

La fermeture transitive de ce graphe est:

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





A retenir :

- ✓ Une matrice M = (a_{ij}) est booléenne si tous ses éléments a_{ij} ne sont que des 0 et des 1.
- ✓ Les matrices associées aux graphes d'ordre n (de n sommets) sont des matrices booléennes d'ordre n (de n lignes et n colonnes).

❖ Recherche de composantes fortement connexes

❑ Forte connexité et graphe réduit (pour les graphes orientés uniquement) :

- un graphe G=(X, E) est dit fortement connexe, si entre chaque paire de sommets (u, v), il existe un chemin allant de u à v et un autre allant de v à u.
- lorsque G n'est pas fortement connexe, on dira qu'il dispose de composantes fortement connexes.

❑ L'algorithme suivant permet de déterminer les composantes fortement connexes d'un graphe donné.

- Son principe consiste à considérer un sommet quelque s ∈ X d'un (+) et d'un (-). Si v non marqué est successeur d'un sommet u marqué d'un (+), on le marque également d'un (+). De même, si un sommet v non marqué est prédécesseur d'un sommet marqué d'un (-), on le marque aussi d'un (-).

Section 2 : Les notions de base sur la théorie des graphes

Tout sommet marqué d'un (+) et (-) appartient à la même composante fortement connexe que S.

- L'algorithme suivant permet de déterminer les composantes fortement connexes d'un graphe non connexe :

(1) Poser i=1

(2) Soit s ∈ X, on marque s d'un signe (+) et (-)

(3) Appliquer la procédure de marquage jusqu'à sa fin

(4) Soit C_i, l'ensemble des sommets marqués d'un (+) et (-). C_i est la i^{ème} composante fortement connexe

Poser X=X\ C_i.

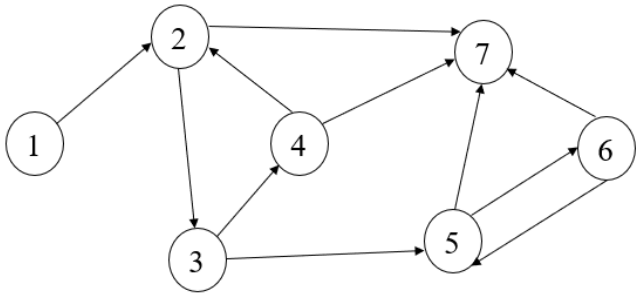
(5) Tester si X=∅, si oui, terminer

Sinon poser i=i+1. Aller en (2).

Section 2 : Les notions de base sur la théorie des graphes

❖ Déterminer les composantes fortement connexes de ce graphe ci-dessous :
Ce graphe possède quatre (4) composantes fortement connexes :

- C_1= **{1}**
- C_2= **{2, 3 ,4}**
- C_3= **{5, 6}**
- C_4= **{7}**



❖Le graphe réduit : soi G = (X, E) un graphe orienté:

❑on appelle graphe réduit de G le graphe G_r dont les sommets C_1, . . . , C_p

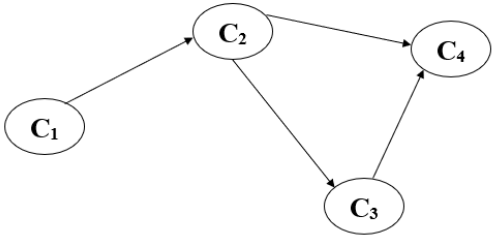
❑sont les composantes fortement connexes de G, et il existe un arc entre C_i

❑et C_j si et seulement s'il existe au moins un arc entre un sommet de G_i et un

❑sommet de G_j dans le graphe G. On vérifie que le graphe G_r est sans circuit

❑(sans boucle).

❖ Le graphe réduit G_r du graphe ci-dessus est le suivant :



A retenir :

✓ Le circuit est appelée une boucle.

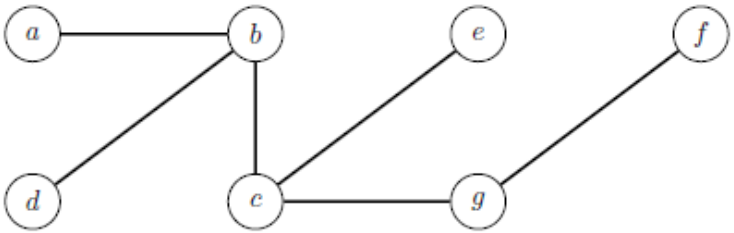
✓ Une source est un sommet qui n'est le successeur de personne (son degré entrant est nul).

✓ Un puits est un sommet qui n'a aucun successeur (son degré sortant est nul).

✓ Un graphe orienté est sans circuit quand il possède une source (respectivement un puits) et que tous ses sous-graphes sont sans circuit.

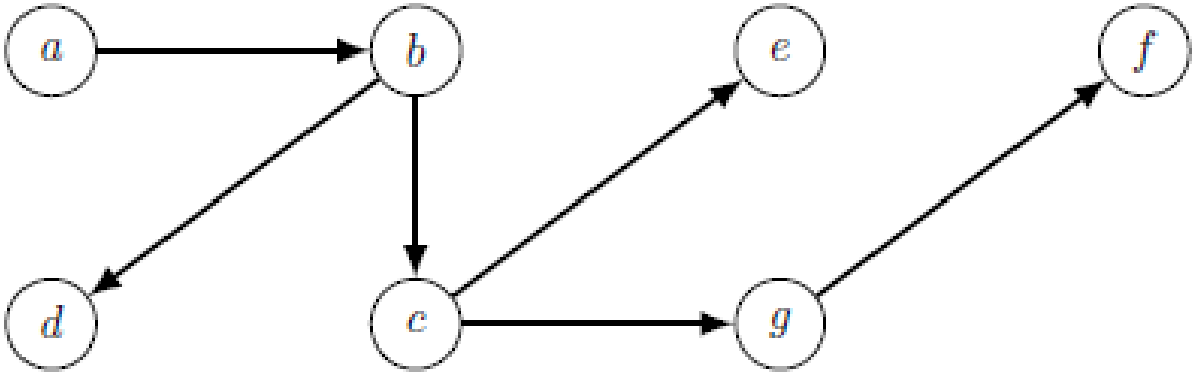
Section 3 : Arbres et Arborescences

- Un arbre est un graphe non orienté G qui vérifie une des conditions équivalentes suivantes :
- G est connexe et acyclique
- G est sans cycle et possède n-1 arêtes
- G est connexe et admet n-1 arêtes
- G est sans cycle, et en ajoutant une arête, on crée un et un seul cycle élémentaire,
- G est connexe, et en supprimant une arête quelconque, il n'est plus connexe,
- il existe une chaîne et une seule entre 2 sommets quelconques de G. Le graphe ci-dessous est un arbre:



Section 3 : Arbres et Arborescences

- On appelle forêt, un graphe dont chaque composante connexe est un arbre.
- Une arborescence est un graphe orienté sans circuit admettant une racine $s_0 \in S$ telle que pour tout autre sommet $s_i \in S$, il existe un chemin unique allant de s_0 vers s_i.
- Si l'arborescence comporte n sommets, alors elle comporte exactement n − 1 arcs. Par exemple, le graphe suivant est une arborescence de racine **a** :

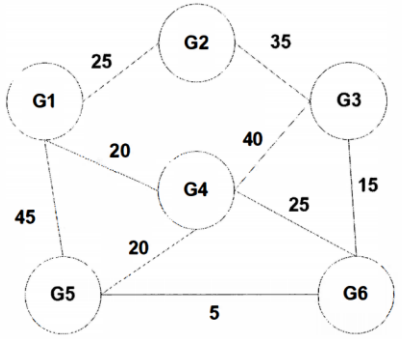


Section 4 : Le parcours de graphe

- Les parcours des graphes permettent de visiter les sommets du graphe à
- partir d’un sommet de départ en suivant les arcs du graphe. Les sommets
- visités sont les sommets qui sont accessibles à partir du sommet de départ
- par un chemin dans le graphe.
- Coût d'un chemin et matrice des longueurs:
- Dans le cadre de la recherche du chemin le plus court, le moins cher ou le
- plus rapide, on doit rajouter des informations. Celles-ci sont appelées de
- manière arbitraire longueurs, sans préciser l'unité. Il peut donc s'agir de
- kilomètres, d'euros, de minutes, de kilos. . .

Section 4 : Le parcours de graphe

- Sur la représentation graphique, on rajoute les longueurs sur le dessin des arcs comme le montre le graphe ci-dessous



- La matrice d'adjacence devient la matrice des longueurs. Les "1" qui représentaient des liens sont alors remplacés par la longueur de l'arc.
- Quand il n'y a pas de lien, on remplace le 0 par +∞, indiquant que pour aller d'un nœud à l'autre, il faudrait un temps/un kilométrage/un coût infini (et donc que ce n'est pas possible) .
- Pour la diagonale, on met des 0, indiquant que pour aller d'un endroit à ce même endroit, on n'a pas à bouger, et donc c'est "gratuit" et immédiat.

Section 4 : Le parcours de graphe

- La matrice des longueurs est donc:

$$\begin{pmatrix} 0 & 25 & \infty & 20 & 45 & \infty \\ 25 & 0 & 35 & \infty & \infty & \infty \\ \infty & 35 & 0 & 40 & \infty & 15 \\ 20 & \infty & 40 & 0 & 20 & 25 \\ 45 & \infty & \infty & 20 & 0 & 5 \\ \infty & \infty & 15 & 25 & 5 & 0 \end{pmatrix}$$

❖ Algorithmes naïfs de recherche de chemin

- Ces premiers algorithmes ne sont pas "intelligents" : ils sont dits naïfs, car ils n'utilisent pas de connaissances sur le problème pour agir. Dans le pire des cas, ils testent tous les chemins possibles pour déterminer s'il existe bien un chemin entre deux nœuds.
- De plus, rien n'indique que le chemin trouvé est bien le plus court. Ils sont cependant faciles à implémenter.
- Voici, les algorithmes naïfs de recherche de chemin:

Section 4 : Le parcours de graphe

- **Parcours en profondeur:**
  - C'est l'algorithme que l'on essaie naturellement dans un labyrinthe : on cherche à avancer le plus possible, et, si on est coincé, on revient à la dernière intersection que l'on a rencontrée et on teste un nouveau chemin.
  - Cet algorithme ne permet pas de déterminer le chemin le plus court, mais simplement de trouver un chemin.
- **Parcours en largeur:**
  - Le parcours en largeur est celui que la police utilise par défaut pour retrouver un coupable. On part du dernier point où celui-ci a été vu, puis on va s'en écarter progressivement en cercles concentriques.

❖ **Algorithmes "intelligents“:**

❑ Les parcours en profondeur et en largeur ne permettent pas de trouver le chemin le plus court, mais juste le premier qui permet de joindre le point de départ au point d'arrivée.

Section 4 : Le parcours de graphe

- D'autres algorithmes existent, qui permettent de déterminer le chemin le plus court, ou au moins un chemin optimisé, et ce sans avoir forcément à tester tous les chemins possibles.Voici, les algorithmes intelligents:

- **Algorithme de Bellman – Ford:**

- L'algorithme de Bellman-Ford permet de trouver le chemin le plus court s'il existe. Il n'est pas le plus optimisé, mais c'est celui qui fonctionne dans le plus de cas. En effet, il accepte des longueurs négatives pour les arcs, et pas seulement des positives.

- **Algorithme de Dijkstra:**

- L'algorithme de Dijkstra est une amélioration de l'algorithme Bellman-Ford. Il ne fonctionne que si toutes les distances sont positives, ce qui est généralement le cas dans les problèmes réels.

Section 4 : Le parcours de graphe

- Il permet de choisir plus intelligemment l'ordre d'application des distances, et surtout au lieu de s'appliquer sur les arcs, il s'applique sur les nœuds, et ne revient jamais en arrière. Ainsi, chaque arc n'est appliqué qu'une seule fois.

Détail de l'algorithme de Dijkstra:

✓ Numérotons les sommets du graphe de 1 à n. Cet algorithme calcule le plus court chemin du sommet 1 à tous les sommets du graphe (il donnera donc la première ligne de la matrice de coût minimum).

✓ On construit un vecteur v = (v(1), v(2), v(3), …, v(n)) ayant n composantes tel que v(i) soit égal à la longueur du plus court chemin allant de 1 au sommet i.

Section 4 : Le parcours de graphe

- ✓ On initialise ce vecteur à (c1,i), c'est-à-dire à la première ligne de la matrice des coûts du graphe.
- ✓ Ensuite, on considère deux (2) ensembles de sommets, S initialisé à $\{1\}$ et $\bar{S}$ son complémentaire dans X, ensemble des sommets du graphe. A chaque pas de l'algorithme, on ajoute à S des sommets jusqu'à ce que S=X de telle sorte que le vecteur v donne à chaque étape le coût minimal des chemins de 1 aux sommets de S.
- ✓ Le tableu ci-dessous donne la description de l'algorithme de Dijsktra

Description de l'algorithme de Dijkstra

initialisations

v=(c1, i) pour i ∈ $\{1,2,\dots,n\}$
S=$\{1\}$; $\overline{S}$ =$\{2,3,\dots,n\}$

Itérations

Tant que $\overline{S} \neq \emptyset$
    Choisir i dans $\overline{S}$ tel que v(i) est minimun
    Retirer i de $\overline{S}$ et l'ajouter à S
    Pour chaque successeur j de i dans $\overline{S}$
        v(j):=min(v(j), v(i)+w(i,j))

Section 4 : Le parcours de graphe

Appliquez l'algorithme de Dijkstra à la matrice w :

$$\begin{bmatrix} 0 & 15 & \infty & \infty & 4 \\ \infty & 0 & \infty & \infty & \infty \\ \infty & 3 & 0 & 2 & \infty \\ 10 & 3 & \infty & 0 & \infty \\ \infty & \infty & 7 & 5 & 0 \end{bmatrix}$$

Initialisation

$S = \{1\}$; $\bar{S} = \{2, 3, 4, 5\}$; $v = (0, 15, \infty, \infty, 4)$

Première itération: $i = 5$ car $v(5) = min(15, \infty, \infty, 4) = 4$

$S = \{1, 5\}$; $\bar{S} = \{2, 3, 4\}$; les successeurs de 5 dans $\bar{S}$ sont 3 et 4, $v(3)$ prend la nouvelle Valeur $min(\infty, v(5) + w(5,3)) = min(\infty, 4 + 7) = 11$, $v(4)$ prend la nouvelle Valeur $min(\infty, v(5) + w(5,4)) = min(\infty, 4 + 5) = 9$, d'où le nouveau vecteur.

Section 4 : Le parcours de graphe

Deuxième itération : $i = 4; v(4) = 9$

$S = \{1, 5, 4\};\ \bar{S} = \{2, 3\};\ v = (0, 12, 11, 9, 4)$

Troisième itération : $i = 3; v(3) = 11$

$S = \{1, 5, 4, 3\};\ \bar{S} = \{2\};\ v = (0, 12, 11, 9, 4)$

Quatrième itération : $i = 2; v(2) = 12$

$S = \{1, 5, 4, 3, 2\};\ \bar{S} = \emptyset;\ v = (0, 12, 11, 9, 4)$

Le chemin minimal de 1 à 4 par exemple est de coût 9. C'est le chemin 1, 5, 4.

Section 4 : Le parcours de graphe

Mini Projet: Recherche du chemin le plus court

Trouver le meilleur chemin reliant deux (2) objets (ou éléments) notamment deux routeurs d'un même réseau informatique est un problème classique en informatique.

Problème: Application de l'agorithme de Dijkstra sur un réseau informatique.

Détails du contenu de votre mini projet:

- Donner le principe de fonctionnement et pseudo-code de cet algorithme intelligent.
- Énoncer le problème proposé
- Donner la représentation graphique du réseau informatique
- Implementer l'algorithme en utilisant l'un de ces langages de votre choix (C, C++, C#, Java et Python).