Cours de Recherche Opérationnelle 2

Enseignant : Afef Bouzaiene
Ecole Nationale des Sciences et Technologies Avancées
Borj Cedria – 2ème année Option S.I.C.

2020-2021

Bibliographie

- Précis de Recherche Opérationnelle, 6ème édition, Robert Faure, Bernard Lemaire, Christophe Picouleau, DUNOD, Paris, 2009.
- Graphes et algorithmes , Michel Gondran et Michel Minoux, 4ème Edition, Lavoisier, 2009.
- Exercices et Problèmes résolus de recherche opérationnelle Tome 1, Roseaux, DUNOD, 2005.
- http://www.lamsade.dauphine.fr/~gabrel/enseignement.php ; Notes de cours « Algorithmique et applications », Virginie Gabrel, CPES 2ème année, 2016-2017.
- http://www.lgi.ecp.fr/~mousseau/Cours/S4/pmwiki/uploads/Main/GraphesAlgoBase.pdf; «Théorie des Graphes, algorithmes de bases », Vincent Mousseau, Ecole Centrale Paris, 2009.

Plan

I. Introduction à la théorie des graphes (Notions élémentaires, Connexité, Représentation de graphes, Algorithmes préliminaires, Cheminement dans les graphes, Programmation dynamique, Flots, Arbres)

II. Notions de complexité des algorithmes et des problèmes d'optimisation

III. Heuristiques et Méta-heuristiques

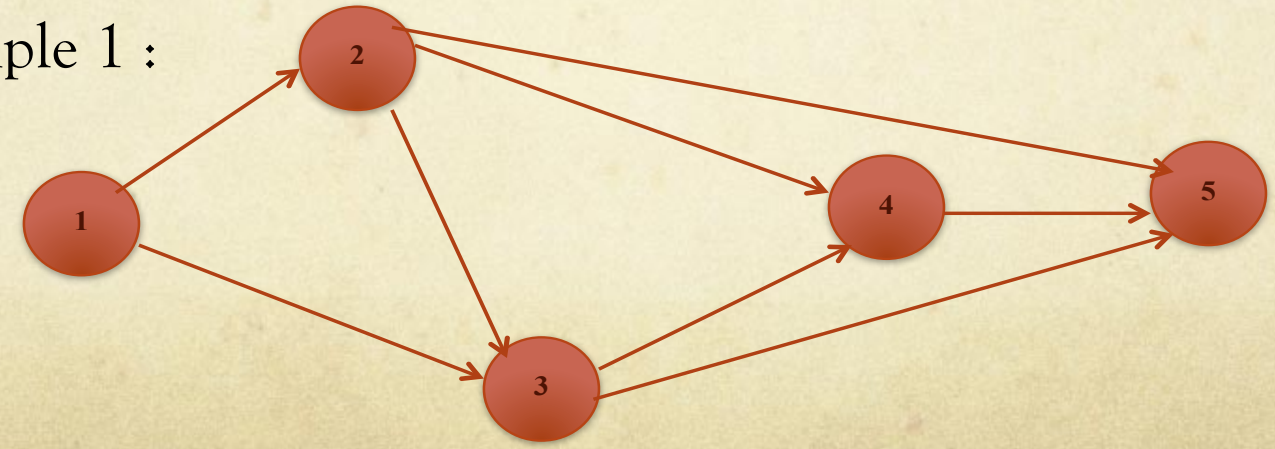
Notions élémentaires

- Un graphe **orienté** G(X,U) est composé de :

- Un ensemble X de sommets ou de nœuds, numérotés de 1 à n. L'ordre de G est n.

- Un ensemble U de couples de sommets ordonnés, u=(i,j), appelés **arcs** (i est l'extrémité initiale ou prédécesseur de j, j est l'extrémité terminale ou successeur de i), card U = m .

- Exemple 1 :



Notions élémentaires

- Une **boucle** est un arc dont l'extrémité initiale est égale à l'extrémité terminale
- Un **p-graphe** est un graphe dans lequel il n'existe pas plus de p arcs de la forme (i,j) entre i et j
- G^{-1} est le graphe inverse obtenu à partir de G en inversant le sens des arcs.
- Un graphe est **complet**, si pour toute paire (i,j), il existe au moins un arc (i,j) ou (j,i)
- Un graphe est **plein**, si pour toute paire (i,j), il existe un arc (i,j) et un arc (j,i)

Notions élémentaires

- L'ensemble des successeurs de i est noté $\Gamma^+(i)$ et l'ensemble des prédécesseurs de i est noté $\Gamma^-(i)$
- Le demi degré extérieur (resp. intérieur) du sommet i, noté $d^+(i)$ (resp. $d^-(i)$) désigne le nombre d'arcs ayant i comme extrémité initiale (resp. terminale). On a :

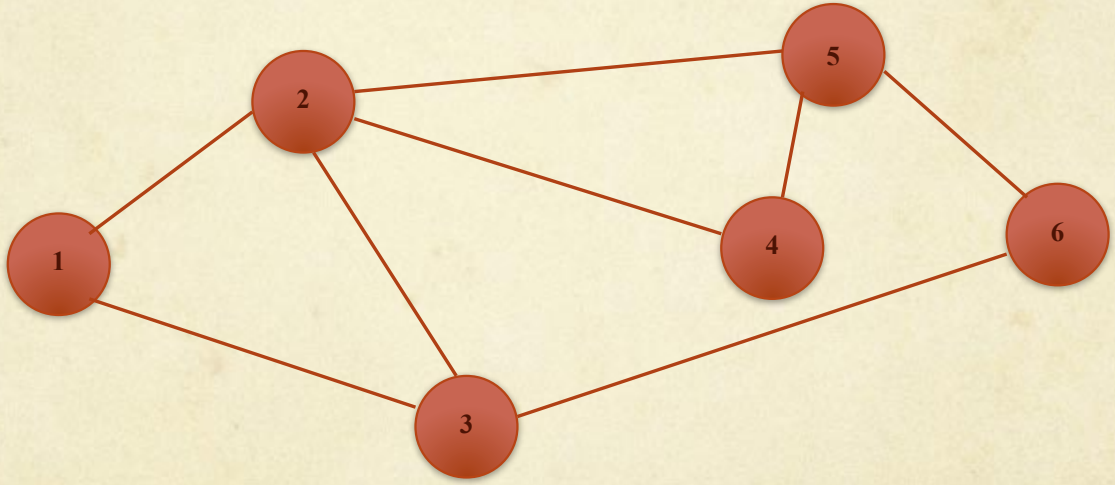
$$d^+(i) = card\ \Gamma^+(i)$$

$$d^-(i) = card\ \Gamma^-(i)$$

- Le degré du sommet i, noté d(i) est le nombre d'arcs ayant i comme extrémité. On a : $\mathbf{d(i)=d^+(i)+d^-(i)}$

Notions élémentaires

- Un graphe **non-orienté** G(X,E) est composé de :
  - Un ensemble X de sommets ou de nœuds, numérotés de 1 à n.
  - Un ensemble de couples de sommets non-ordonnés, e=(i,j), appelés **arêtes**
- Exemple 2:



- $\Gamma(i) = \{j \in X : (i,j) \in E\}$ est le voisinage de i et le degré de i est donné par $d(i) = card\ \Gamma(i)$

Notions élémentaires

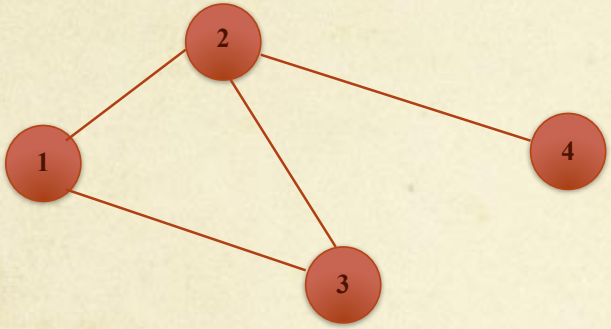
- Un **multigraphe** est un graphe non-orienté où plusieurs arêtes peuvent exister entre deux sommets i et j
- Un graphe est dit **simple** s'il est sans boucle, et s'il n'existe pas plus d'une arête entre deux sommets i et j
- Deux arcs (arêtes) sont **adjacent(e)s** s'ils ont au moins une extrémité en commun.
- Une **Clique** (non-orienté) est un sous-ensemble $C \subseteq X$ tel que deux sommets quelconques de C sont reliés par une arête. Un sommet isolé constitue une clique.
- Un **Stable** (non-orienté) est un sous-ensemble $S \subseteq X$ tel que deux sommets quelconques de S ne sont pas reliés par une arête. Un sommet isolé constitue un stable.

Notions élémentaires

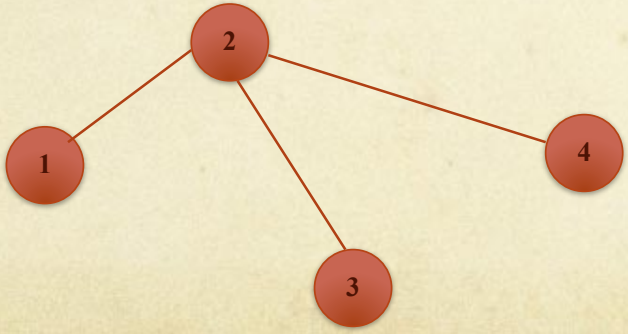
- Un **sous-graphe engendré par un sous-ensemble de sommet** $G_A \subset G$ est un graphe dont les sommets sont dans A de X, et dont les arcs éléments de E ont leurs deux extrémités dans A
- Un **graphe partiel engendré par un sous-ensemble d'arcs** $V \subset E$; dont les sommets sont dans X et les arêtes sont dans V.
- Soient A dans X et V dans E, un **sous-graphe partiel engendré par A et V** est le graphe partiel de G_A engendré par V

Notions élémentaires (exple2)

- Sous - Graphe engendré par A ={1,2,3,4} :

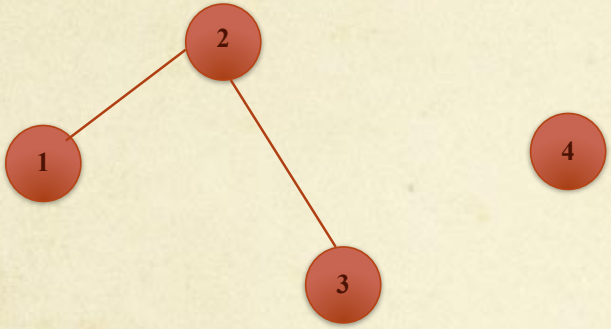


- Graphe partiel engendré par V= {(1,2), (2,3), (2,4)}

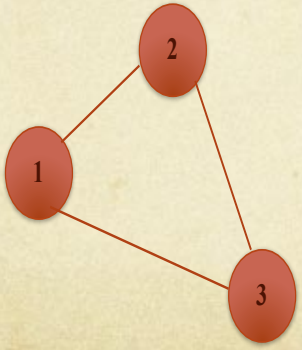


Notions élémentaires (exple2)

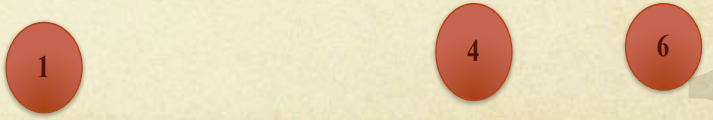
- Sous - Graphe partiel de A={1,2,3,4} et V={(1,2), (2,3)}



- Clique



Stable



Notions élémentaires

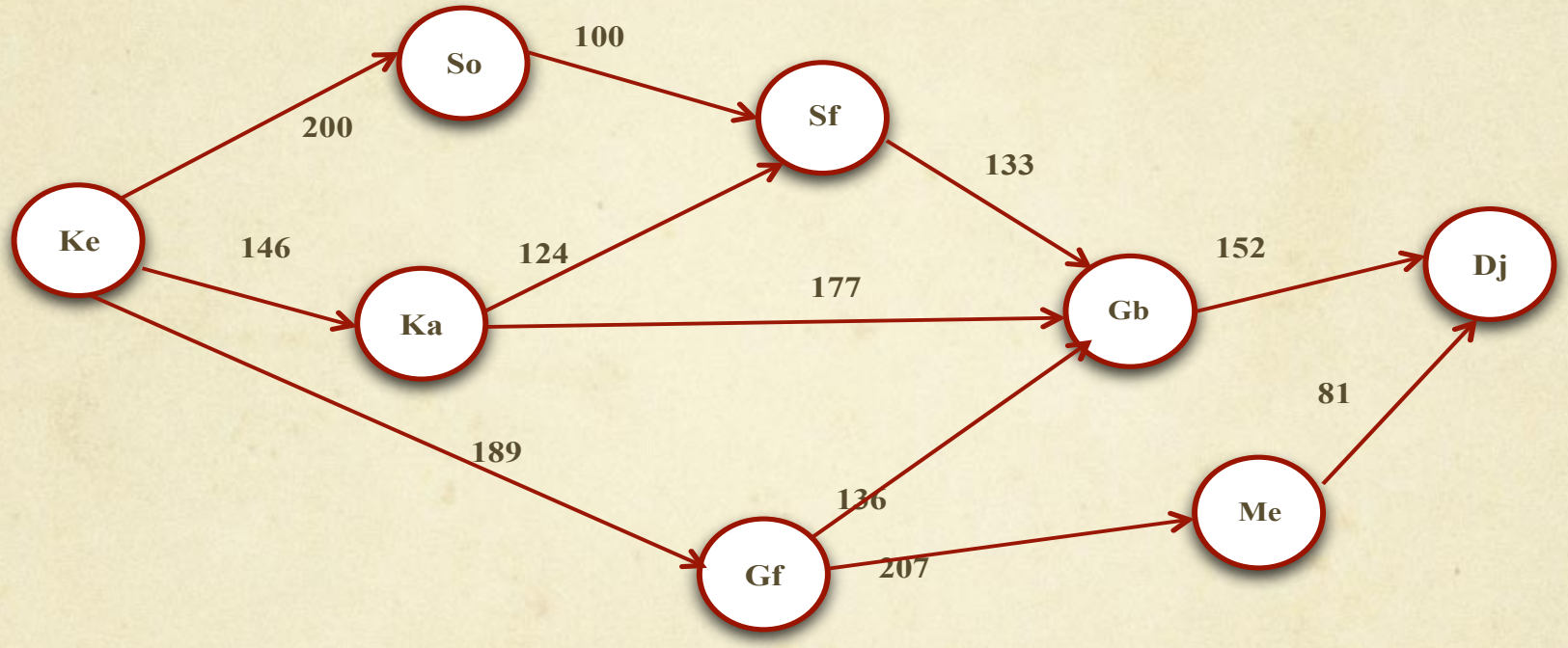
- Un graphe orienté peut être transformé en un graphe non-orienté en enlevant le sens des arcs. Et un graphe non-orienté peut être transformé en un graphe orienté en remplaçant chaque arête (i,j) par deux arcs (i,j) et (j,i).
- Un graphe (orienté ou non) est dit **valué** quand ses arcs (arêtes) et/ou ses sommets sont dotés d'un poids (ou longueur)

Quelques problèmes de graphes

Exemple 3: Comment faire pour aller le plus rapidement possible du Kef à Djerba ?

Ville de Départ	Ville d' Arrivée	Durée du trajet
Le Kef	Sousse	3h 20 min
Le Kef	Kairouan	2h 26 min
Le Kef	Gafsa	3h 09 min
Sousse	Sfax	1h 40 min
Kairouan	Sfax	2h 04 min
Kairouan	Gabès	2h 57 min
Sfax	Gabès	2h 13 min
Gafsa	Gabès	2h 16 min
Gafsa	Medenine	3h 27 min
Gabès	Djerba	2h 32 min
Medenine	Djerba	1h 21 min

Quelques problèmes de graphes (exple3)



=> Le problème consiste à chercher **le plus court chemin** de la ville de départ jusqu'à la ville d'arrivée.

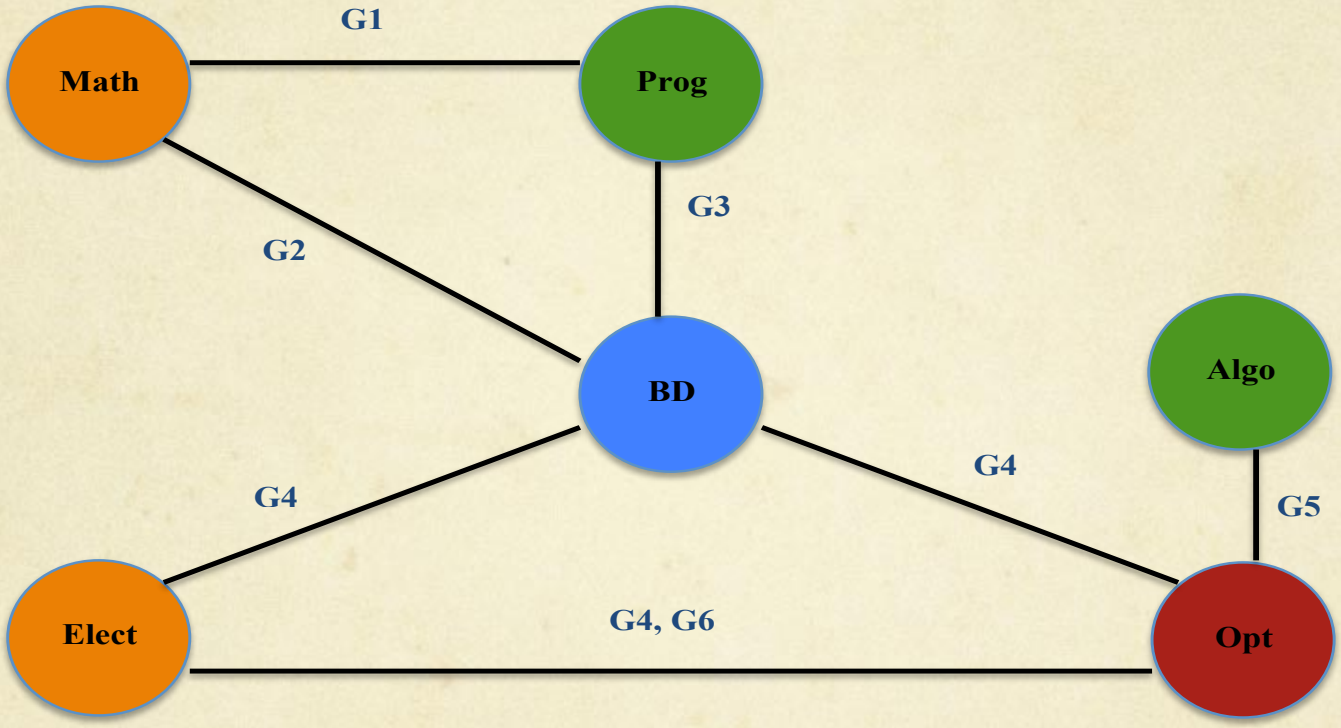
Quelques problèmes de graphes

Exemple 4: Comment organiser une session d'examen la plus courte possible pour 6 groupes d'étudiants (chaque examen dure 3 heures) ?

Discipline	Groupes
Mathématiques	G1, G2
Programmation	G1, G3
Bases de Données	G2, G3, G4
Optimisation	G4, G5, G6
Electronique	G4, G6
Algorithmique	G5

- les disciplines sont à représenter par des sommets. Relier par une arête les disciplines dont les examens ne peuvent se dérouler simultanément. Colorier par deux couleurs distinctes deux sommets reliés par une arête.

Quelques problèmes de graphes (exple4)



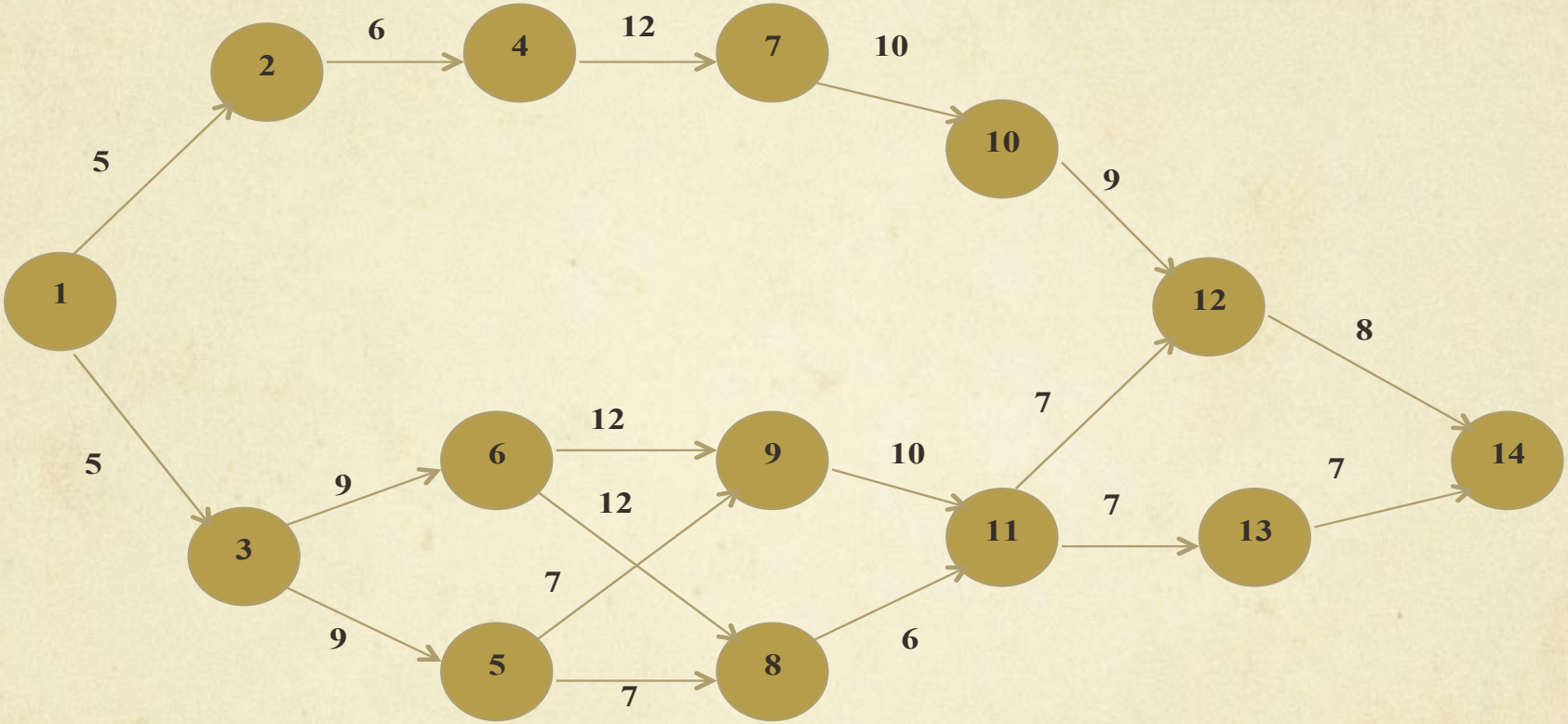
Le problème est alors de colorier tous les sommets du graphe en utilisant le moins de couleurs possible (**le nombre chromatique**) => Problème de coloration de graphes

Quelques problèmes traités par les graphes

Exemple 5 : Ordonnancement de tâches dans un projet (construction d'un stade)

Tâche	Description	Durée	Successeurs
1	Terrassements	5	2, 3
2	Construction des fondations	6	4
3	Voirie, réseaux divers	9	5, 6
4	Plancher principal	12	7
5	Cloisonnement des vestiaires	7	8, 9
6	Electrification des gradins	12	8, 9
7	Pose du toit	10	10
8	Eclairage du stade	6	11
9	Installation des gradins	10	11
10	Construction de la billetterie	9	12
11	Voirie secondaire	7	12, 13
12	Signalétique	8	14
13	Finition des vestiaires, pelouse et accessoires sportifs	7	14

Quelques problèmes de graphes (exple5)

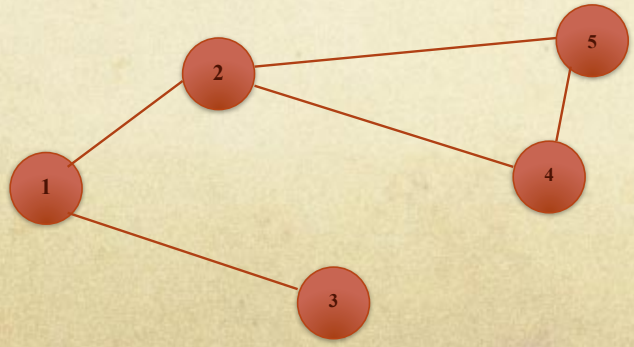


=> Le problème consiste à trouver **le chemin le plus long** dans un graphe, ce qui permettra de déterminer la durée totale du projet puis les dates de début au plus tôt et au plus tard de chaque tâche.

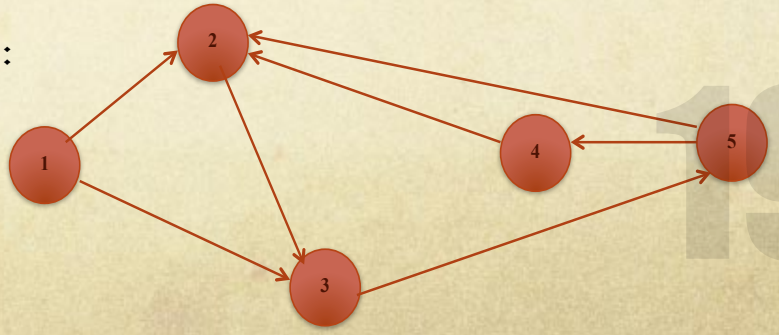
Connexité

- Une **chaîne** dans un graphe (non-orienté) est une suite de sommets reliés par des arêtes. Sa longueur est le nombre d'arêtes qu'elle comporte. Une chaîne dans l'ex.6 : (1, 2, 4, 5)
- Un **chemin** dans un graphe (orienté) est une suite de sommets reliés par des arcs ayant le même sens. Sa longueur est le nombre d'arcs qu'il comporte. Un chemin dans l'ex.7 : (1,2,3,5,4)
- Un **cycle** (resp. **circuit**) est une chaîne (resp. un chemin), dont le dernier sommet est le premier (les extrémités coïncident). (2,4,5,2) est un cycle dans l'ex.6 et (2,3,5,2) est un circuit dans l'ex.7

Ex 6:



Ex 7:



Connexité

Une chaîne, un chemin, un cycle ou un circuit est :

- **élémentaire**, si les sommets qu'il comporte sont tous distincts (sauf les extrémités).
- **simple**, si les arêtes (les arcs) qu'il comporte sont tous distincts.
- **hamiltonien**, s'il passe une fois et une seule par chaque sommet du graphe.
- **eulérien**, s'il passe une fois et une seule par chaque arc du graphe.
- **préhamiltonien**, s'il passe au moins une fois par chaque sommet du graphe.
- **préeulérien**, s'il passe au moins une fois par chaque arc du graphe.

Connexité

- Dans l'ex.6 : (1,2,4,5) chaîne élémentaire ; (1,2,4,5,2) chaîne simple (non élémentaire) ; une chaîne (un chemin etc.) élémentaire est nécessairement simple.

  (1,2,4,5,2,1,3) est une chaîne préeulérienne et préhamiltonienne

  (3,1,2,4,5) est hamiltonienne ; (3,1,2,4,5,2) est eulérienne

- Dans l'ex.7 : (1,2,3,5,4) est un chemin hamiltonien (non eulérien)

  (1,2,3,5,4,2) est un chemin préhamiltonien

Connexité

- On note $\mu(i,j)$ (resp. $\mu[i,j]$) est une chaîne (resp. un chemin), reliant deux sommets d'un graphe, i et j
- Un graphe est dit **connexe** si $\forall (i,j) \in X \times X, \exists\ \mu(i,j)$
- Un graphe est dit **fortement connexe** si

$$\forall (i,j) \in X \times X, \exists\ \mu[i,j]\ et\ \mu[j,i]$$

- Exemples de graphes particuliers :

- graphe bi-parti $X = X_1 \cup X_2$ et $\forall e = (i,j);\ i \in X_1 et\ j \in X_2$ ou $j \in X_1 et\ i \in X_2$

- graphe planaire ; c'est un graphe que l'on peut représenter dans le plan sans que les arcs (ou les arêtes) ne se coupent.

Représentation de graphes

- Soit G=(X,U) un 1-graphe, la **matrice d'adjacence** A, composée de n × n éléments a_{ij}, définis comme suit :

$$\begin{cases} a_{ij} = 1 \ si \ (i,j) \in U \\ a_{ij} = 0 \ \text{si} \ non \end{cases}$$

- Soit G=(X,U) un graphe sans boucle, la **matrice d'incidence sommets-arcs** A, composée de n × m éléments a_{ij} définis comme suit (chaque colonne correspond à un arc et chaque ligne à un sommet) : Si u=(i,j) est un arc de U alors la colonne de u a tous ses termes nuls sauf a_{iu}=+1 et a_{ju}= −1

- Soit G=(X,E) un graphe non-orienté sans boucle, une **matrice d'incidence sommets-arêtes** (G non-orienté sans boucle); tous les éléments de la colonne u sont nuls sauf a_{iu}=1 et a_{ju}=1

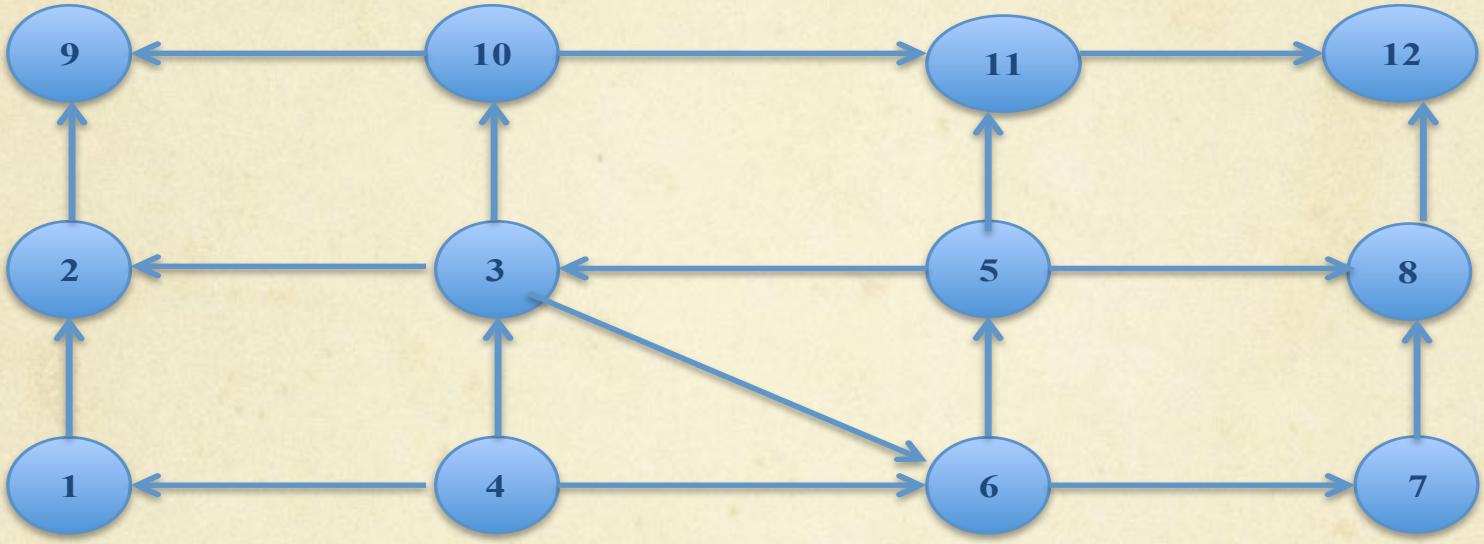
Représentation de graphes

- Pour un graphe orienté G(X,U), on considère le tableau des successeurs (dictionnaire des suivants) pour chacun des sommets, constitué des deux colonnes; i : sommet ; $\Gamma^{+}(i)$.
- Pour un graphe orienté G(X,U), on considère le tableau des prédécesseurs (dictionnaire des précédents) pour chacun des sommets, constitué des deux colonnes; i : sommet ; $\Gamma^{-}(i)$.

Représentation de graphes

- Exercice :

Soit le graphe orienté ci-dessous (exemple 8) :



1/ Représenter le graphe par la matrice d'adjacence puis par la matrice d'incidence sommets-arcs .

2/ Représenter le graphe par le dictionnaire des suivants, puis par le dictionnaire des précédents.

Représentation de graphes

1/ Matrice d'adjacence :

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	1	0	0	0
3	0	1	0	0	0	1	0	0	0	1	0	0
4	1	0	1	0	0	1	0	0	0	0	0	0
5	0	0	1	0	0	0	0	1	0	0	1	0
6	0	0	0	0	1	0	1	0	0	0	0	0
7	0	0	0	0	0	0	0	1	0	0	0	0
8	0	0	0	0	1	0	1	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	1	0	1	0
11	0	0	0	0	0	0	0	0	0	0	0	1
12	0	0	0	0	0	0	0	0	0	0	0	0

Représentation de graphes

1/ Matrice d'incidence sommets-arcs : (les cases vides représentent des zéros)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	1					-1												
2	-1	1	-1															
3			1	1	1		-1		-1									
4						1	1	1										
5									1	1	1	-1						
6				-1				-1				1	1					
7													-1	1				
8										-1				-1	1			
9		-1														-1		
10					-1											1	1	
11											-1						-1	1
12															-1			-1

Représentation de graphes

1/ voir Annexe 1 ; 2/ Dictionnaire des suivants :

i	$\Gamma^+(i)$
1	2
2	9
3	2,6,10
4	1,3,6
5	3,8,11
6	5,7
7	8
8	12
9	$\varnothing$
10	9,11
11	12
12	$\varnothing$

Représentation de graphes

2/ (suite) Dictionnaire des précédents :

i	$\Gamma^{-}(i)$
1	4
2	1,3
3	4,5
4	$\varnothing$
5	6
6	3,4
7	6
8	5,7
9	2,10
10	3
11	5,10
12	8,11

Algorithmes préliminaires

- **Détection de circuits dans un graphe orienté**
- **Tri topologique d'un graphe orienté sans circuits**
- Parcours dans les graphes (ne seront pas détaillés dans ce cours)

Algorithme de détection de circuits dans un graphe orienté

- Idée principale :

Aucun circuit ne passe par un sommet n'ayant pas de successeur
(x / $\Gamma^+(x) = \emptyset$)

Un sommet y ne peut pas faire partie de circuit(s) si tous ses successeurs x vérifient $\Gamma^+(x) = \emptyset$

Eliminer les chemins qui ne sont pas des circuits de la manière suivante :

⇒ Marquer et supprimer progressivement les sommets qui n'ont pas de successeurs. Si tous les sommets sont marqués alors le graphe ne contient pas de circuit.

Remarque : Un sommet marqué est considéré comme supprimé de la liste des successeurs

Algorithme de détection de circuits dans un graphe orienté

- Début
- Représenter le graphe par son dictionnaire des suivants.
- Tant qu'il existe un sommet non-marqué qui n'a pas de successeurs

  - Choisir et marquer ce sommet

  - Marquer ce sommet dans les listes de successeurs de tous les sommets
- Fin Tant que
- Si tous les sommets sont marqués alors il n'existe pas de circuit ; sinon il existe au moins un circuit passant par les sommets non-marqués
- Fin Si
- Fin

Exercice : Appliquer l'algorithme de détection de circuits sur le graphe de l'exemple 8 ainsi que celui de l'exemple 3

Tri topologique dans graphe orienté

- Un tri topologique d'un graphe acyclique orienté est un ordre de visite des sommets tel qu'un sommet soit toujours visité avant ses successeurs :

Soit G(X,U) orienté avec X={0,1,2,…, n-1}. Un ordre topologique sur G est une bijection o: X-> X telle que pour tout arc (u,v) de G, o(u)<o(v). L'ordre o est représenté par le tableau [o(0); o(1); …; o(n-1)] qui est une ligne.

- **Un graphe est sans circuit si et seulement si il admet un ordre topologique.**

Tri topologique dans graphe orienté

- Tout graphe sans circuit peut être mis en ordre par rang croissant : rang(x) , $x \in X$ qui est le nombre d'arcs du plus long chemin ayant x pour extrémité terminale.

- Les C_i contiennent les sommets de rang i
- Un tri topologique est une numérotation compatible avec les rangs
- Une numérotation n: x → IN est compatible avec les rangs si et seulement si :

$$\forall x, y \in X,\ si\ x \in C_i et\ y \in C_j avec\ i < j\ alors\ n(x) < n(y)$$

Mise en ordre du graphe par rang croissant

- début
  - Représenter G par son dictionnaire des précédents $\Gamma^{-1}(x)$
  - $M \leftarrow \emptyset$
  - $i \leftarrow 0$
  - tant que $M \neq X$ faire
    - Soit $C_i = \{x \in X,\ \text{t.q. } \Gamma^{-1}(x) = \emptyset\}$
    - $M \leftarrow M \cup C_i$
    - Supprimer tous les sommets de M de la liste des prédécesseurs de tous les sommets ($\rightarrow sous - graphe G_{X \setminus M}$)
    - $i \leftarrow i + 1$
  - fin tant que
- fin

•Numérotation des sommets :

•Début

•i←0; j←0;

•Tant qu'il existe un sommet non numéroté faire

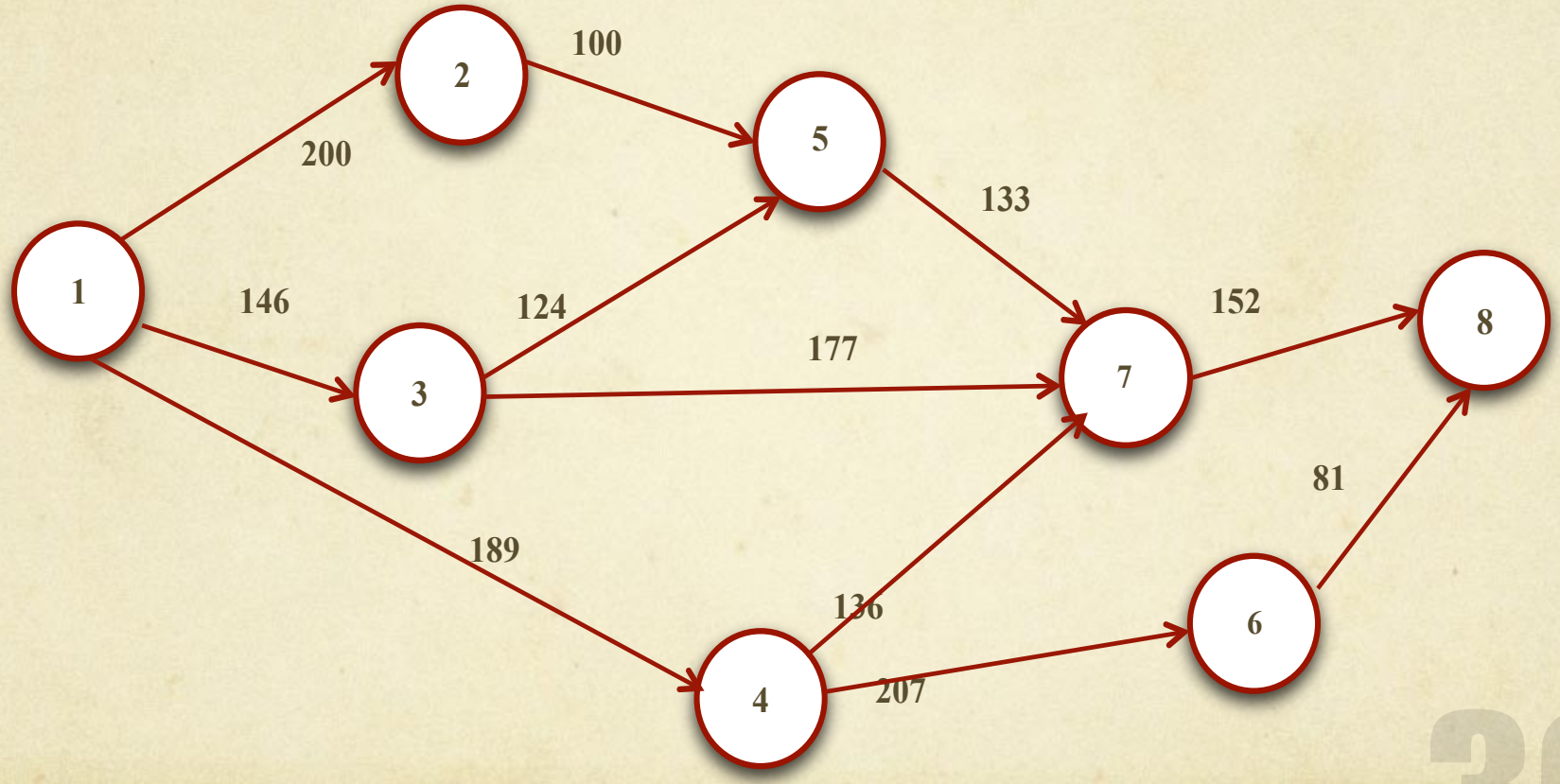
- Numéroter arbitrairement les sommets de C_i de j+1 à j+ Card (C_i)
- j←j+Card(C_i);

i← i+1;

•FinTantque

Tri topologique dans graphe orienté

- Application de l'algorithme de tri topologique à l'exemple 3 :



Chemins optimaux dans les graphes

- Etant donné un graphe orienté G(X,U), où l'on associe à chaque arc u=(i,j) un nombre réel l(i,j) appelée longueur ou poids de u ;
- Le Problème de recherche du **chemin de valeur minimale (chemin le plus court) :**

- à partir d'un sommet origine i_0 vers un sommet quelconque i;

consiste à déterminer parmi tous les chemins reliant le sommet origine et le sommet terminal, le chemin μ^* dont la longueur totale:

$$l(\mu^*) = \sum_{u \in \mu^*} l(u)$$, soit **minimale**

Condition nécessaire et suffisante au Problème du chemin de valeur optimale

- Le Problème du chemin le plus court (resp. le plus long) entre deux sommets, dans un graphe orienté valué, admet une solution si et seulement s'il n'existe pas dans le graphe un circuit de valeur strictement négative (resp. positive), appelé circuit absorbant, pouvant être atteint à partir du sommet origine.

- **Indications de preuve :** (pour le P.C.C.)

- **(sens =>)** Si cette condition nécessaire est vérifiée, il existe toujours un chemin de longueur minimale qui soit élémentaire.

- **(sens <=)** En effet, lorsque tous les circuits du graphe pouvant être atteint à partir du sommet origine ont une longueur strictement positive, tout plus court chemin est nécessairement élémentaire.

Graphes à longueurs positives

- Chemin le plus court à partir d'un sommet i_0 vers tout autre sommet du graphe => aucun circuit absorbant n'existe puisque les longueurs sont positives

- **Algorithme de Dijkstra (1959):**

$\lambda(i_0) \leftarrow 0$
$\lambda(i) \leftarrow +\infty \; \forall i \in X \setminus \{i_0\}$
$p(i) \leftarrow i \; \forall i \in X$
$S \leftarrow \{i_0\}$
$i \leftarrow i_0$
TantQue $S \neq X$ Faire
  Pour tout $j \in \Gamma^+(i) \setminus S$ Faire
    Si $\lambda(j) > \lambda(i) + l_{ij}$ alors
      $\lambda(j) \leftarrow \lambda(i) + l_{ij}$
      $p(j) \leftarrow i$
    FinSi
  FinPour
  Sélectionner $i \in X \setminus S$ tel que $\lambda(i) = \min_{j \in X \setminus S} \lambda(j)$
  $S \leftarrow S \cup \{i\}$
FinTantQue

Graphes à longueurs positives

- **Algorithme de Dijkstra :** (application à l'exemple 3)

$$\lambda(1)=0;\ \lambda(i)=\infty\ ,\forall i\in X\setminus\{1\}\ ;p(i)=i;\ \forall i\in X;S=\{1\};\ i=1;$$
$$S\neq X;j\in\{2,3,4\};\lambda(2)=200;p(2)=1;\lambda(3)=146;p(3)=1;$$
$$\lambda(4)=189;p(4)=1;$$
$$i\in X\setminus\{1\};\lambda(i)=\min_{j\in X\setminus\{1\}}\lambda(j)=146=\lambda(3)\ ;i=3;S=\{1,3\}$$

$$S\neq X;j\in\{5,7\};\lambda(5)=270;p(5)=3;\lambda(7)=323;p(7)=3;$$
$$i\in X\setminus\{1,3\};\lambda(i)=\min_{j\in X\setminus\{1,3\}}\lambda(j)=189=\lambda(4)\ ;i=4;S=\{1,3,4\};$$

$$S\neq X;j\in\{6,7\};\lambda(6)=396;p(6)=4;\lambda(7)=323;$$
$$i\in X\setminus\{1,3,4\};\lambda(i)=\min_{j\in X\setminus\{1,3,4\}}\lambda(j)=200=\lambda(2)\ ;i=2;S=\{1,2,3,4\};$$

Graphes à longueurs positives

- **Algorithme de Dijkstra :** (application à l'exemple 3)

$$S \neq X; j \in \{5\}; \lambda(5) = 270;$$
$$i \in X \setminus \{1,2,3,4\}; \lambda(i) = \min_{j \in X \setminus \{1,2,3,4\}} \lambda(j) = 270 = \lambda(5) \ ; i = 5; S = \{1,2,3,4,5\};$$

$$S \neq X; j \in \{7\}; \lambda(7) = 323;$$
$$i \in X \setminus \{1,2,3,4,5\}; \lambda(i) = \min_{j \in X \setminus \{1,2,3,4,5\}} \lambda(j) = 323 = \lambda(7) \ ; i = 7; S = \{1,2,3,4,5,7\};$$

$$S \neq X; j \in \{8\}; \lambda(8) = 475; p(8) = 7;$$
$$i \in X \setminus \{1,2,3,4,5,7\}; \lambda(i) = \min_{j \in X \setminus \{1,2,3,4,5,7\}} \lambda(j) = 396 = \lambda(6) \ ; i = 6;$$
$$S = \{1,2,3,4,5,6,7\};$$

Graphes à longueurs positives

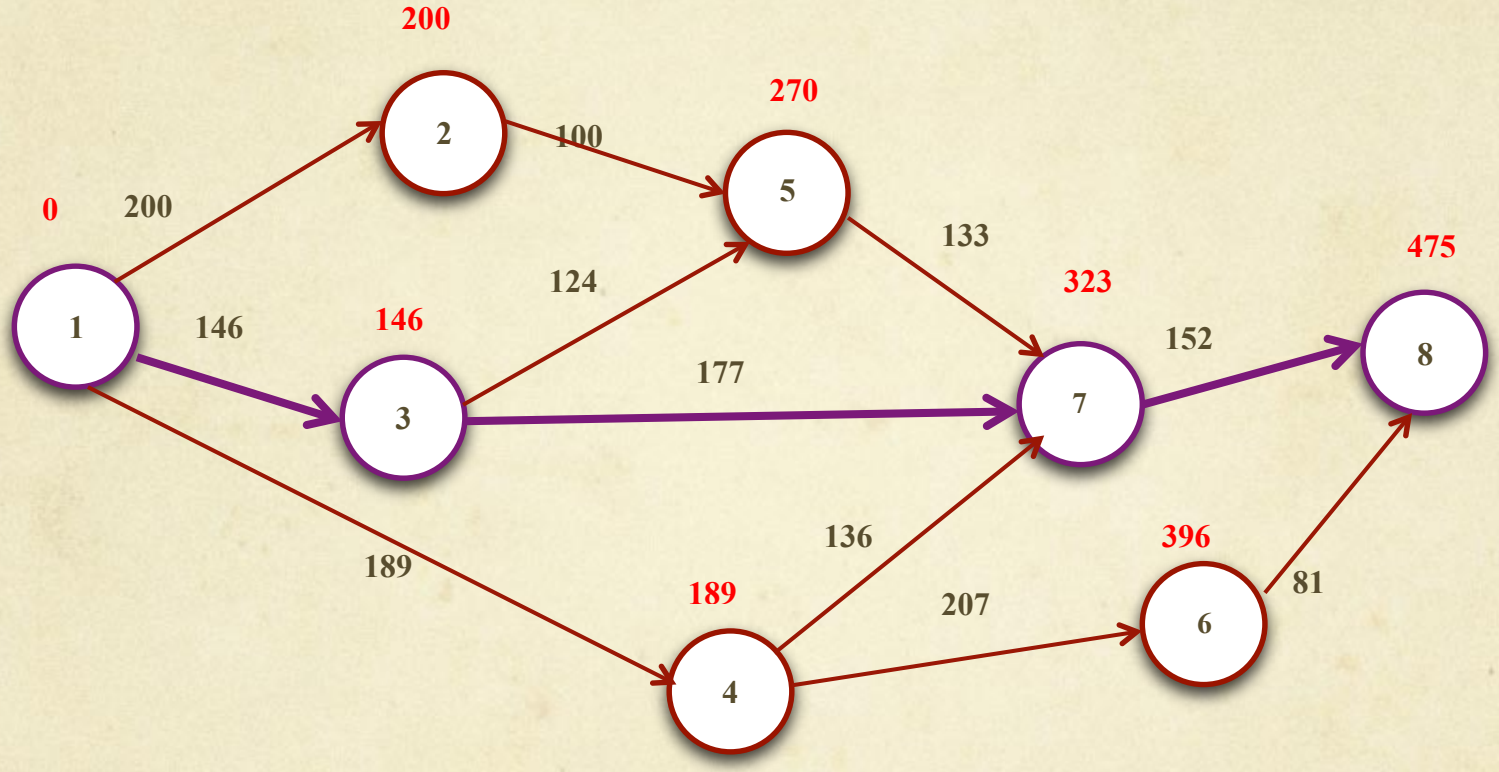
- **Algorithme de Dijkstra :** (application à l'exemple 3)

$$S \neq X; j \in \{8\}; \lambda(8) = 475;$$
$$i \in X \setminus \{1,2,3,4,5,6,7\}; \lambda(i) = \min_{j \in X \setminus \{1,2,3,4,5,6,7\}} \lambda(j) = 475 = \lambda(8) \; ; i = 8;$$
$$S = X$$

$$p(8) = 7; p(7) = 3; p(3) = 1 \Rightarrow$$
$$\mu^* = (1,3,7,8) \; ; l(\mu^*) = 475$$

Graphes à longueurs positives

- **Algorithme de Dijkstra :** (application à l'exemple 3)



Lorsque les plus courtes distances ont été calculées de 1 vers 2, 3, 4 et 5, il est inutile pour calculer la plus courte distance de 1 à 7 de remonter au-delà des prédécesseurs de 7.
=> On utilise le principe de **la programmation dynamique => tout sous-chemin d'un chemin optimal est optimal**

Graphes à longueurs quelconques

Algorithme de Bellman (1958) : (entre i_0 et i quelconque du graphe)

$k \leftarrow 0;\ \ \lambda^0(i_0) \leftarrow 0;$

Pour tout $i \in X \setminus \{i_0\}$ *faire* :

$\lambda^0(i\ ) \leftarrow +\infty;$

FinPour

($\lambda^k(i)$ *correspond à la valeur du plus court chemin entre* i_0 *et* i *en utilisant* k *arcs au plus*)

Répéter

$k \leftarrow k+1;$

Pour tout $i \in X$ *faire*

$$\lambda^k(i) \leftarrow \min\{\lambda^{k-1}(i), \min_{j \in \Gamma_i^{-1}}\{\lambda^{k-1}(j) + l_{ji}\}\};$$

FinPour

Tant que $((k \le n)$ *et* $(\exists i \in X / \lambda^k(i) \neq \lambda^{k-1}(i)))$

Si $(\exists i \in X / \lambda^n(i) \neq \lambda^{n-1}(i))$ *alors*

Il existe un circuit de longueur négative passant par i

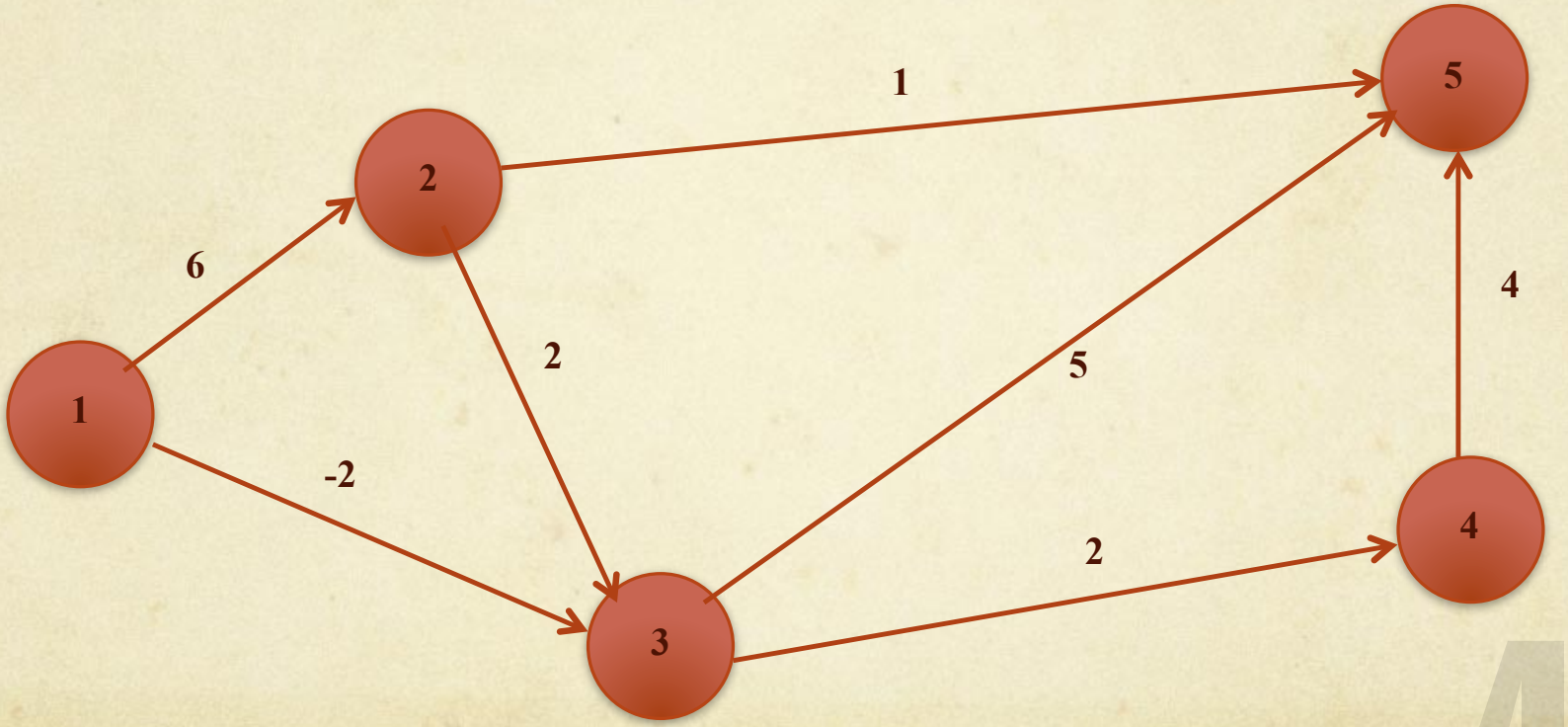
Sinon

$\lambda^k(i)$ *correspond à la valeur du plus court chemin entre* i_0 *et* i.

FinSi

Graphes à longueurs quelconques

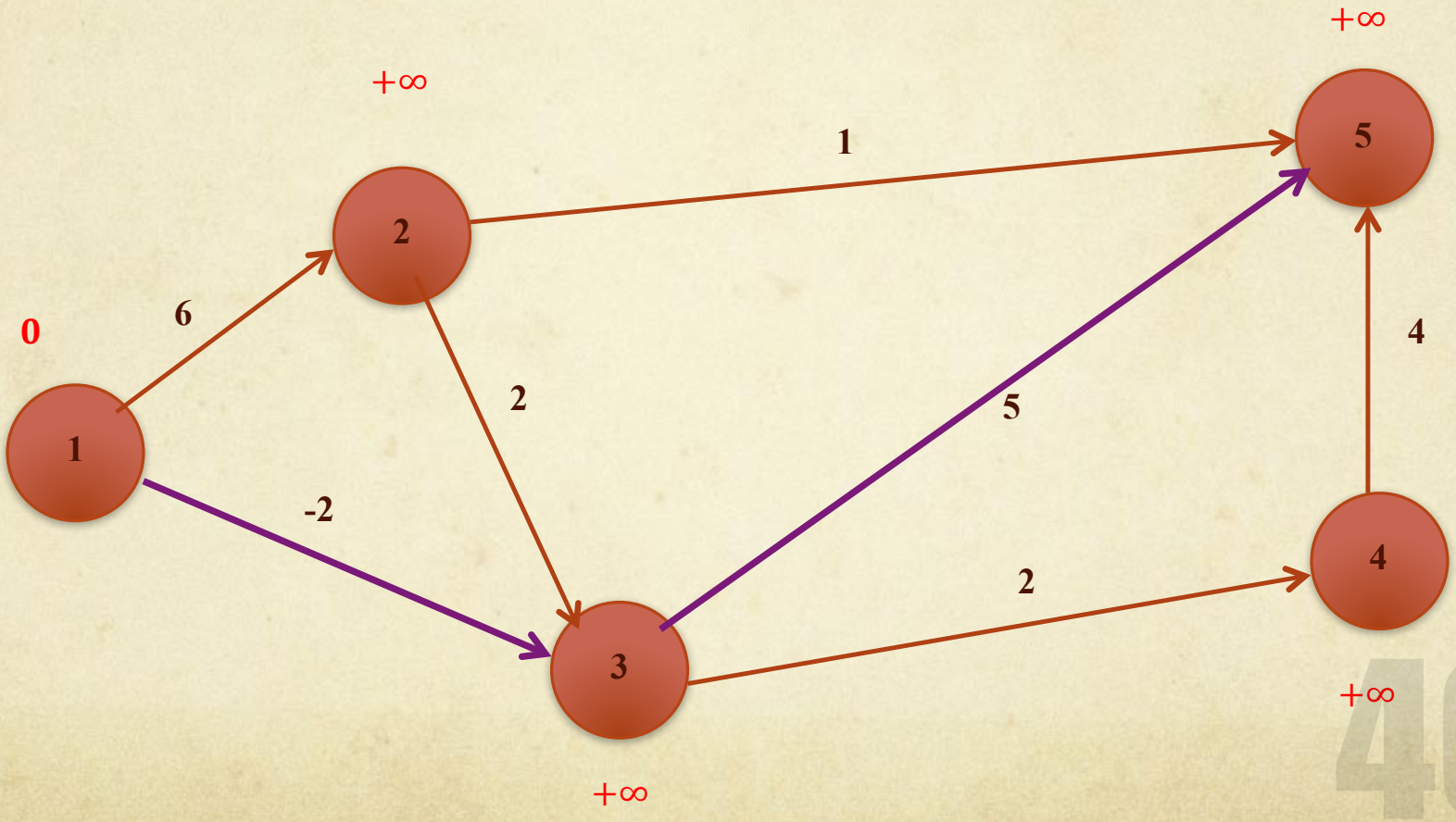
- Exemple 9 :



Graphe à longueurs quelconques

- **Algorithme de Bellman :** (application à l'exemple 9)

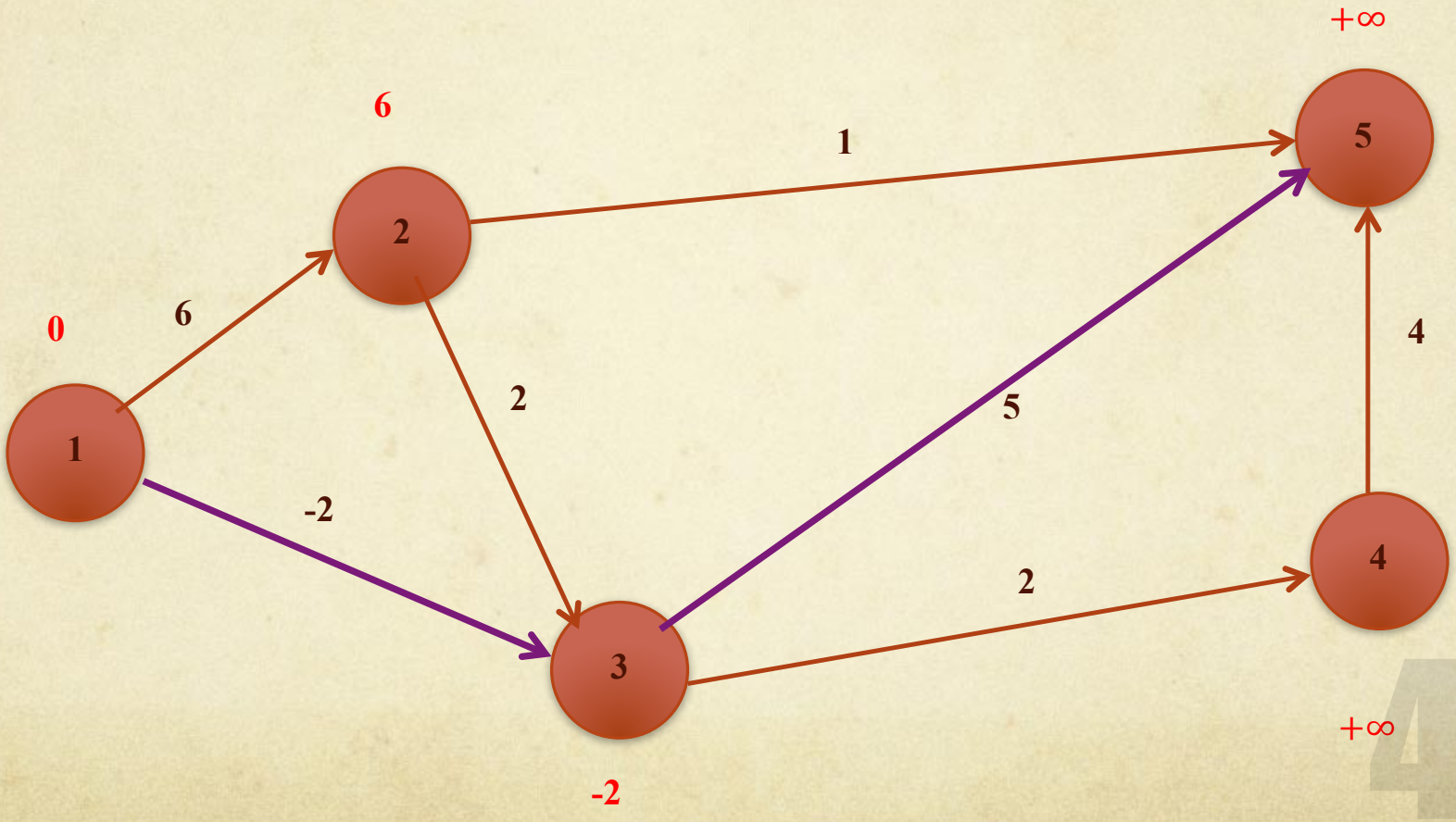
k=0



Graphe à longueurs quelconques

- **Algorithme de Bellman :** (application à l'exemple 9)

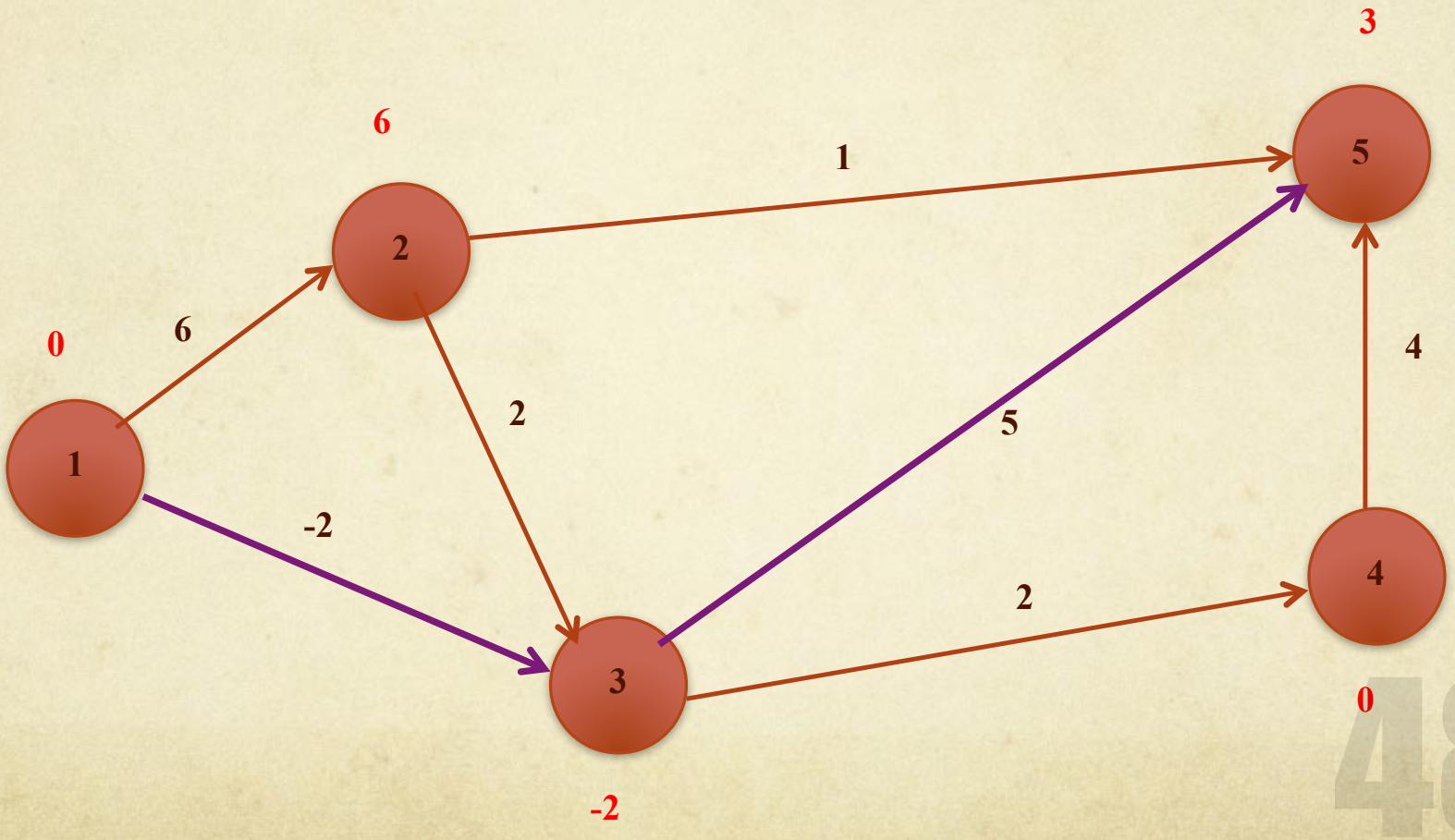
k=1



Graphe à longueurs quelconques

- **Algorithme de Bellman :** (application à l'exemple 9)

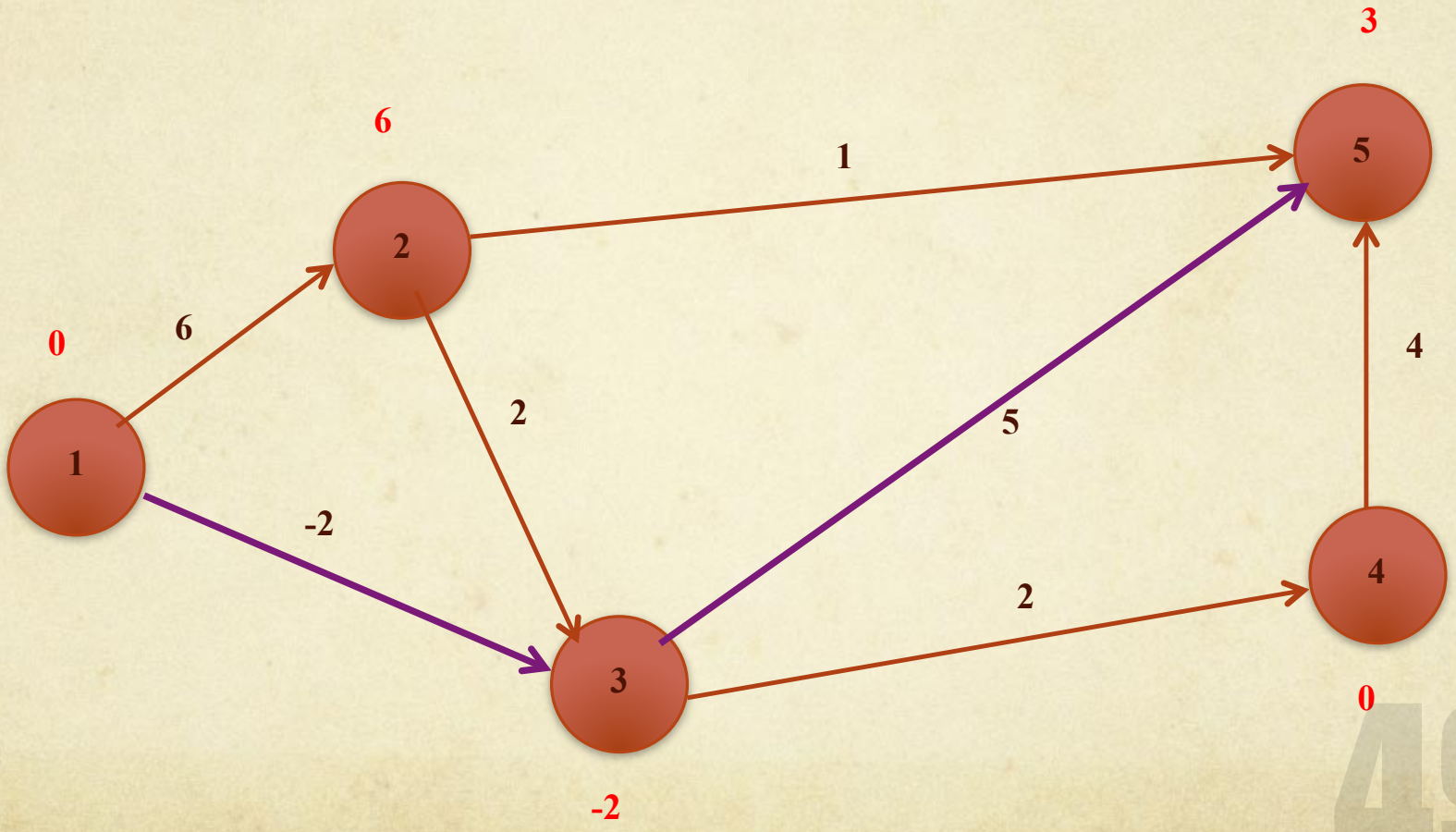
k=2



Graphe à longueurs quelconques

- **Algorithme de Bellman :** (application à l'exemple 9)

k=3

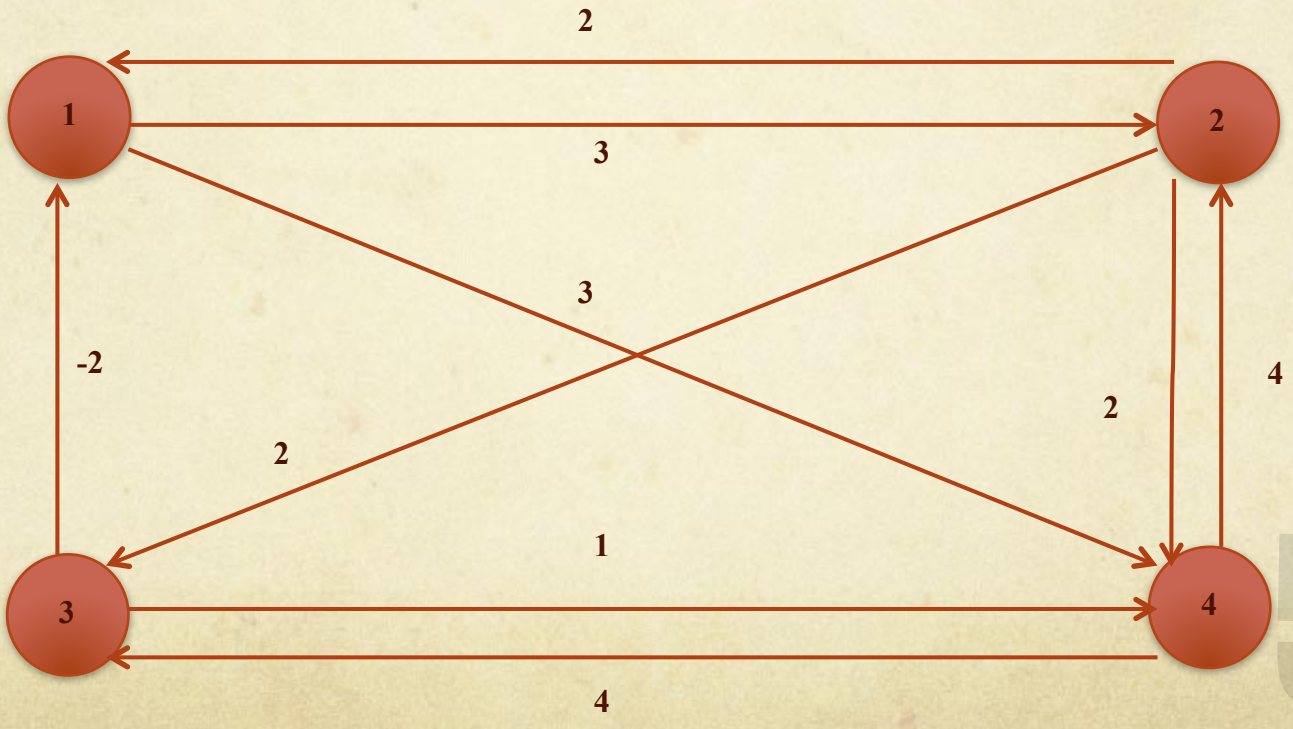


Arrêt

Graphes à longueurs quelconques

- Chemin le plus court entre toutes les paires de sommets d'un graphe représenté par sa matrice des distances l_{ij} si l'arc (i,j) existe et $+\infty$ sinon; $l_{ii}=0$.

- Exemple 10 :



Graphes à longueurs quelconques

Algorithme de Floyd (1962) : (chemin le plus court entre toutes les paires i et j du graphe)

Pour tout $k \in X$ *faire* :

Pour tout $i \in X$ *faire* :

Si $(l_{ik} \neq +\infty)$ *ou* $(l_{ik} + l_{ki} < 0)$ *alors* :

Si $(l_{ik} + l_{ki} < 0)$ *alors* :

Stop

(*Il existe un circuit de longueur négative passant par i et k*)

FinSi

Sinon

Pour tout $(j \in X)$ *faire* :

$l_{ij} \leftarrow \min\{l_{ij}; l_{ik} + l_{kj}\};$

FinPour

FinSi

FinPour

FinPour

Graphes à longueurs quelconques

Algorithme de Floyd (1962) : (Application à l'exemple 10)

Initialisation

(i,j)	1	2	3	4
1	0	3	+∞	3
2	2	0	2	2
3	-2	+∞	0	1
4	+∞	4	4	0

Graphes à longueurs quelconques

Algorithme de Floyd (1962) : (Application à l'exemple 10)

k=1;

i=1;

$l_{11}=0 \neq +\infty \Rightarrow l_{11} + l_{11} = 0 \Rightarrow$ (Sinon) $j=1 ; l_{11} = \min \{l_{11} ; l_{11} + l_{11}\} = 0 ;$

$j=2 ; l_{12} = \min \{l_{12} ; l_{11} + l_{12}\} = 3 ; j=3 ; l_{13} = \min \{l_{13} ; l_{11} + l_{13}\} = +\infty ;$
$j=4 ; l_{14} = \min \{l_{14} ; l_{11} + l_{14}\} = 3$

i=2;

$l_{21}=3 \neq +\infty \Rightarrow l_{21} + l_{12} = 5 \Rightarrow$ (Sinon) $j=1 ; l_{21} = \min \{l_{21} ; l_{21} + l_{11}\} = 2 ;$

$j=2 ; l_{22} = \min \{l_{22} ; l_{21} + l_{12}\} = 0 ; j=3 ; l_{23} = \min \{l_{23} ; l_{21} + l_{13}\} = 2 ;$
$j=4 ; l_{24} = \min \{l_{24} ; l_{21} + l_{14}\} = 2$

Graphes à longueurs quelconques

Algorithme de Floyd (1962) : (Application à l'exemple 10)

i=3;

l_{31}=-2 ≠+∞ => l_{31} + l_{13} =+∞ => (Sinon) j=1 ; l_{31} = min {l_{31} ; l_{31} + l_{13} } = -2 ;

j=2 ; l_{32} = min {l_{32} ; l_{31} + l_{12} } =1 ; j=3 ; l_{33} = min {l_{33} ; l_{31} + l_{13} } =0 ;
j=4 ; l_{34} = min {l_{34} ; l_{31} + l_{14} } =1

i=4;

L_{41}=+∞ et l_{41} + l_{14} = +∞ => k= 2 … (exercice à compléter)

Graphes à longueurs quelconques

Algorithme de Floyd (1962) : (Application à l'exemple 10)

k=1

(i,j)	1	2	3	4
1	0	3	+∞	3
2	2	0	2	2
3	-2	1	0	1
4	+∞	4	4	0

Graphes à longueurs quelconques

Algorithme de Floyd (1962) : (Application à l'exemple 10)

k=2

(i,j)	1	2	3	4
1	0	3	5	3
2	2	0	2	2
3	-2	1	0	1
4	6	4	4	0

Graphes à longueurs quelconques

Algorithme de Floyd (1962) : (Application à l'exemple 10)

k=3

(i,j)	1	2	3	4
1	0	3	5	3
2	0	0	2	2
3	-2	1	0	1
4	2	4	4	0

Graphes à longueurs quelconques

Algorithme de Floyd (1962) : (Application à l'exemple 10)

k=4

(i,j)	1	2	3	4
1	0	3	5	3
2	0	0	2	2
3	-2	1	0	1
4	2	4	4	0