Recherche opérationnelle (I)
Ordonnancement

bruno.colombel@univ-amu.fr

DUT Informatique
IUT d'Aix-Marseille
Site d'Arles

2019–2020

Sommaire

Sommaire

Définition

Recherche opérationnelle

Approche scientifique pour la résolution de problèmes de gestion de systèmes complexes

Définition

Recherche opérationnelle

Approche scientifique pour la résolution de problèmes de gestion de systèmes complexes

Wikipedia

Operations research, operational research, or simply OR, is the use of mathematical models, statistics and algorithms to aid in decision-making

Recherche opérationnelle

Science du « comment mieux faire avec moins »

Des outils pour

- rationaliser
- simuler
- optimiser
- planifier

l'architecture et le fonctionnement des systèmes industriels et économiques.

Recherche opérationnelle

Science du « comment mieux faire avec moins »

Des outils pour

- rationaliser
- simuler
- optimiser
- planifier

l'architecture et le fonctionnement des systèmes industriels et économiques.

Des modèles pour analyser des situations complexes

Recherche opérationnelle

Science du « comment mieux faire avec moins »

Des outils pour

- rationaliser
- simuler
- optimiser
- planifier

l'architecture et le fonctionnement des systèmes industriels et économiques.

Des modèles pour analyser des situations complexes

Permet aux décideurs de faire des choix efficaces et robustes

Recherche opérationnelle

Approche quantitative pour produire les meilleures décisions

- Une discipline à la croisée des mathématiques et de l'informatique

Recherche opérationnelle

Approche quantitative pour produire les meilleures décisions

- Une discipline à la croisée des mathématiques et de l'informatique
  - prolongement de l'algorithmique

Recherche opérationnelle

Approche quantitative pour produire les meilleures décisions

- Une discipline à la croisée des mathématiques et de l'informatique
  - prolongement de l'algorithmique
  - manipulant des structures plus élaborées : graphes, polyèdres...

Recherche opérationnelle

Approche quantitative pour produire les meilleures décisions

- Une discipline à la croisée des mathématiques et de l'informatique
  - prolongement de l'algorithmique
  - manipulant des structures plus élaborées : graphes, polyèdres...
  - domaine d'application de la théorie de la complexité algorithmique

Recherche opérationnelle

Approche quantitative pour produire les meilleures décisions

- Une discipline à la croisée des mathématiques et de l'informatique
  - prolongement de l'algorithmique
  - manipulant des structures plus élaborées : graphes, polyèdres...
  - domaine d'application de la théorie de la complexité algorithmique
- Une boite à outils de méthodes pour aborder sainement et sereinement les problèmes d'optimisation

Les outils de RO-AD

- aident à trouver

Les outils de RO-AD

- aident à trouver
  - une solution où l'homme n'en trouvait pas

Les outils de RO-AD

- aident à trouver
  - une solution où l'homme n'en trouvait pas
  - une solution sur des problèmes nouveaux où l'homme n'a aucune expérience

Les outils de RO-AD

- aident à trouver
  - une solution où l'homme n'en trouvait pas
  - une solution sur des problèmes nouveaux où l'homme n'a aucune expérience
  - plusieurs solutions là où l'homme n'en envisageait qu'une

Les outils de RO-AD

- aident à trouver
  - une solution où l'homme n'en trouvait pas
  - une solution sur des problèmes nouveaux où l'homme n'a aucune expérience
  - plusieurs solutions là où l'homme n'en envisageait qu'une
- aident à juger de la qualité d'une solution

Les outils de RO-AD

- aident à trouver
  - une solution où l'homme n'en trouvait pas
  - une solution sur des problèmes nouveaux où l'homme n'a aucune expérience
  - plusieurs solutions là où l'homme n'en envisageait qu'une
- aident à juger de la qualité d'une solution
- aident à confirmer / justifier des décisions

Problème du voyageur de commerce

- Un voyageur de commerce, basé à Toulon, doit visiter ses clients à travers la France.
- Il souhaite effectuer la tournée la plus courte possible.

Problème du voyageur de commerce

- Un voyageur de commerce, basé à Toulon, doit visiter ses clients à travers la France.
- Il souhaite effectuer la tournée la plus courte possible.

Recherche du plus court chemin

Transport — Logistique

Transport

- de marchandises
- des entrepôts vers les clients
- coûts de transport, distance sur les arcs
- trouver le meilleur plan de distribution

Recherche opérationnelle

Face à un problème pratique de décision :

Recherche opérationnelle

Face à un problème pratique de décision :

- Aspects mathématiques (contraintes, objectifs, simplifications)

Recherche opérationnelle

Face à un problème pratique de décision :

- Aspects mathématiques (contraintes, objectifs, simplifications)
- Modélisation

Recherche opérationnelle

Face à un problème pratique de décision :

- Aspects mathématiques (contraintes, objectifs, simplifications)
- Modélisation
- Analyse des modèles et résolution

Recherche opérationnelle

Face à un problème pratique de décision :

- Aspects mathématiques (contraintes, objectifs, simplifications)
- Modélisation
- Analyse des modèles et résolution
  - étude de complexité : que peut-on espérer pour le temps de résolution imparti ?

Recherche opérationnelle

Face à un problème pratique de décision :

- Aspects mathématiques (contraintes, objectifs, simplifications)
- Modélisation
- Analyse des modèles et résolution
  - étude de complexité : que peut-on espérer pour le temps de résolution imparti ?
  - mise au point d'algorithmes

Recherche opérationnelle

Face à un problème pratique de décision :

- Aspects mathématiques (contraintes, objectifs, simplifications)
- Modélisation
- Analyse des modèles et résolution
  - étude de complexité : que peut-on espérer pour le temps de résolution imparti ?
  - mise au point d'algorithmes
- Implémentation et analyse des résultats (valider par rapport à la demande)

Recherche opérationnelle

Face à un problème pratique de décision :

- Aspects mathématiques (contraintes, objectifs, simplifications)
- Modélisation
- Analyse des modèles et résolution
  - étude de complexité : que peut-on espérer pour le temps de résolution imparti ?
  - mise au point d'algorithmes
- Implémentation et analyse des résultats (valider par rapport à la demande)
- Déploiement des solutions (Intégration logicielle)

Recherche Opérationnelle : Outils

Programmation linéaire

coût min / profit max $\qquad \max / \min \quad c_1 x_1 + c_2 x_2 \cdots c_n x_n$

quantités produites $\qquad x_1,\ x_2,\ \ldots,\ x_n \geqslant 0$

Recherche Opérationnelle : Outils

Programmation linéaire

coût min / profit max $\quad \max / \min \quad c_1x_1 + c_2x_2 \cdots c_nx_n$

satisfaire la demande $\quad a_1x_1 + a_2x_2 \cdots a_ncx_n \geqslant b_1$

quantités produites $\quad x_1,\ x_2,\ \ldots,\ x_n \geqslant 0$

Recherche Opérationnelle : Outils

Programmation linéaire

coût min / profit max	$\max / \min \quad c_1x_1 + c_2x_2 \cdots c_nx_n$
satisfaire la demande	$a_1x_1 + a_2x_2 \cdots a_ncx_n \geqslant b_1$
avec des ressources limitées	$a'_1x_1 + a'_2x_2 \cdots a'_ncx_n \leqslant b'_1$
quantités produites	$x_1,\ x_2,\ \ldots,\ x_n \geqslant 0$

Recherche Opérationnelle : Outils

Optimisation Combinatoire

- Trouver la meilleure solution parmi un nombre fini mais très grand de choix

Optimisation Combinatoire

- Trouver la meilleure solution parmi un nombre fini mais très grand de choix
- Un problème d'OC se caractérise par :
  - La présence de choix, à faire parmi un ensemble fini d'alternatives

Optimisation Combinatoire

- Trouver la meilleure solution parmi un nombre fini mais très grand de choix
- Un problème d'OC se caractérise par :
  - La présence de choix, à faire parmi un ensemble fini d'alternatives
  - Une notion de coût, ou de gain, ou de perte

Optimisation Combinatoire

- Trouver la meilleure solution parmi un nombre fini mais très grand de choix
- Un problème d'OC se caractérise par :
  - La présence de choix, à faire parmi un ensemble fini d'alternatives
  - Une notion de coût, ou de gain, ou de perte
  - La nécessité de faire globalement les bons choix, de manière à optimiser la valeur objectif

Optimisation Combinatoire

- Trouver la meilleure solution parmi un nombre fini mais très grand de choix
- Un problème d'OC se caractérise par :
  - La présence de choix, à faire parmi un ensemble fini d'alternatives
  - Une notion de coût, ou de gain, ou de perte
  - La nécessité de faire globalement les bons choix, de manière à optimiser la valeur objectif
- Exemples : emplois du temps. . .

Recherche Opérationnelle : Outils

Graphes

Graphes

- meilleur chemin de i à j

Recherche Opérationnelle : Outils

Graphes

- meilleur chemin de i à j
- meilleurs parcours

Recherche Opérationnelle : Outils

Graphes

- meilleur chemin de i à j
- meilleurs parcours
- Représentation de réseaux

Recherche Opérationnelle : Outils

Graphes

- meilleur chemin de i à j
- meilleurs parcours
- Représentation de réseaux
- Ordonnancement

Recherche Opérationnelle : Outils

Graphes

- meilleur chemin de i à j
- meilleurs parcours
- Représentation de réseaux
- Ordonnancement
- Compatibilité de produits

Recherche Opérationnelle : Outils

Et bien d'autres outils encore ...

Sommaire

Exemple

Métros :

Ligne 1 La Rose <-> La Timone

Ligne 2 Bougainville <-> Sainte Marguerite Dromel

Tramway :

Ligne 68 Noailles <-> Saint Pierre

Graphes orientés

Définition

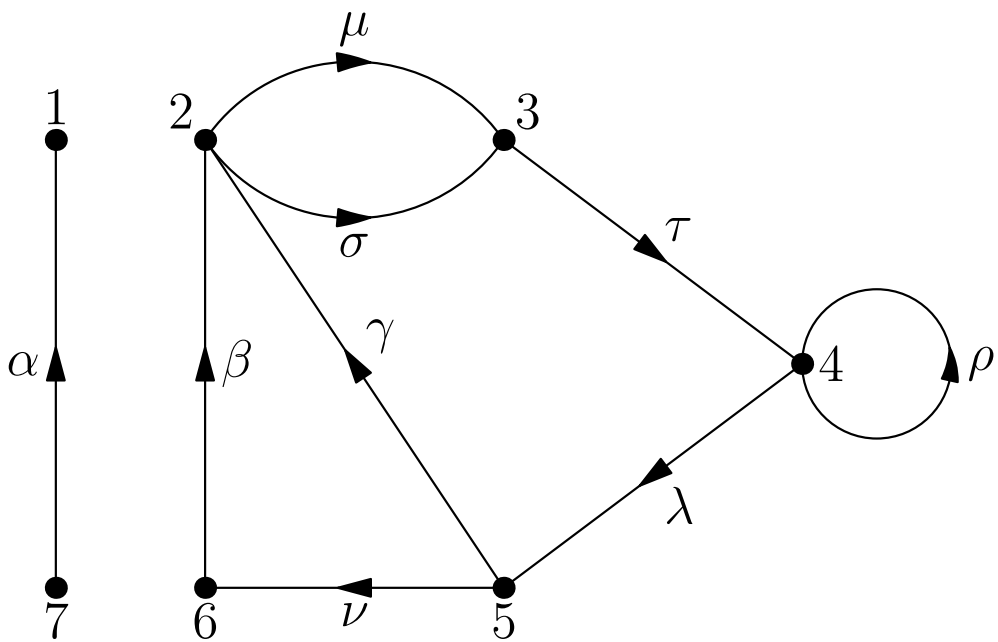
Un *graphe orienté* $\mathscr{G}$ est constitué :

- d'un ensemble fini $\mathcal{S}$ (les sommets)
- d'un ensemble fini $\mathcal{A}$ (les arêtes)
- d'une application $\delta : \mathcal{A} \to \mathcal{S}^2$ qui à une arrête associe 2 sommets

Un graphe $\mathscr{G}$ est désigné par le triplet :

$$\mathscr{G} = (\mathcal{S}; \mathcal{A}; \delta)$$

Graphes orientés



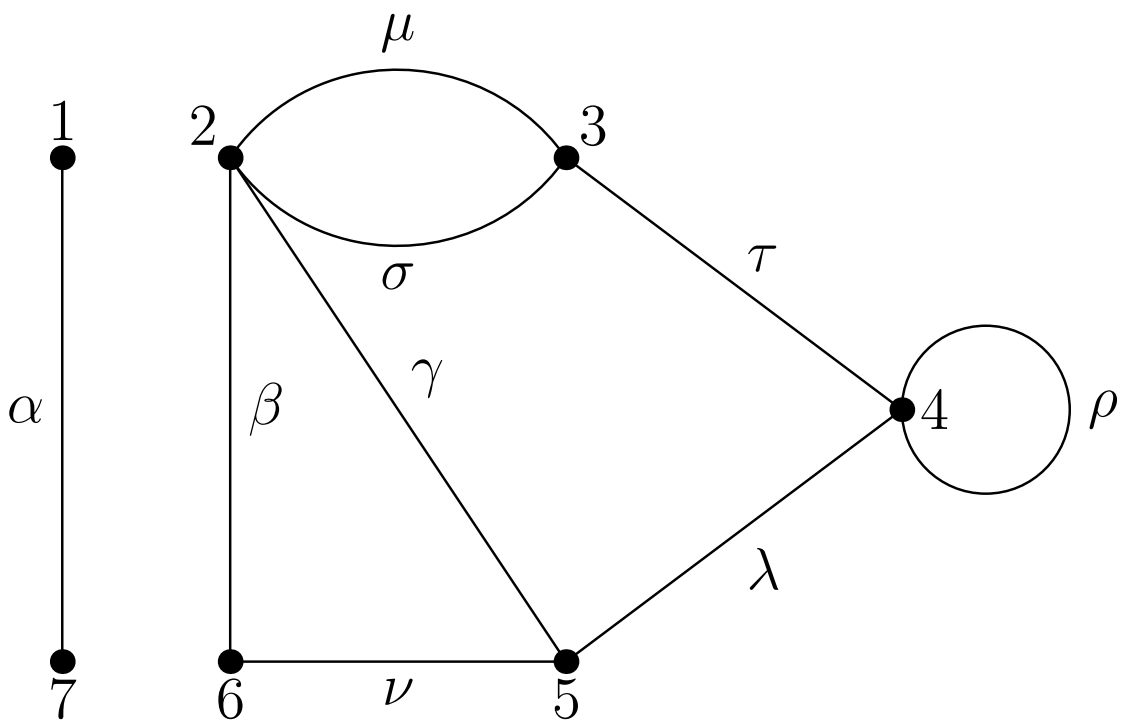
Graphes non-orientés

Définition

Un *graphe non-orienté* $\mathscr{G}$ est constitué :

- d'un ensemble fini $\mathcal{S}$ (les sommets)
- d'un ensemble fini $\mathcal{A}$ (les arêtes)
- d'une application $\delta : \mathcal{A} \to \mathcal{S}_1 \cup \mathcal{S}_2$ où $\mathcal{S}_i$ est l'ensemble des parties à k éléments de $\mathcal{S}$.

Graphes non-orientés



Degré d'un sommet

Définition

Soit s le sommet d'un graphe non-orienté. Le *degré* de s est égal au nombre d'arêtes dont s est une extrémité.

Degré d'un sommet

Définition

Soit s le sommet d'un graphe non-orienté. Le *degré* de s est égal au nombre d'arêtes dont s est une extrémité.

Attention, les boucles comptent double !

Degré d'un sommet

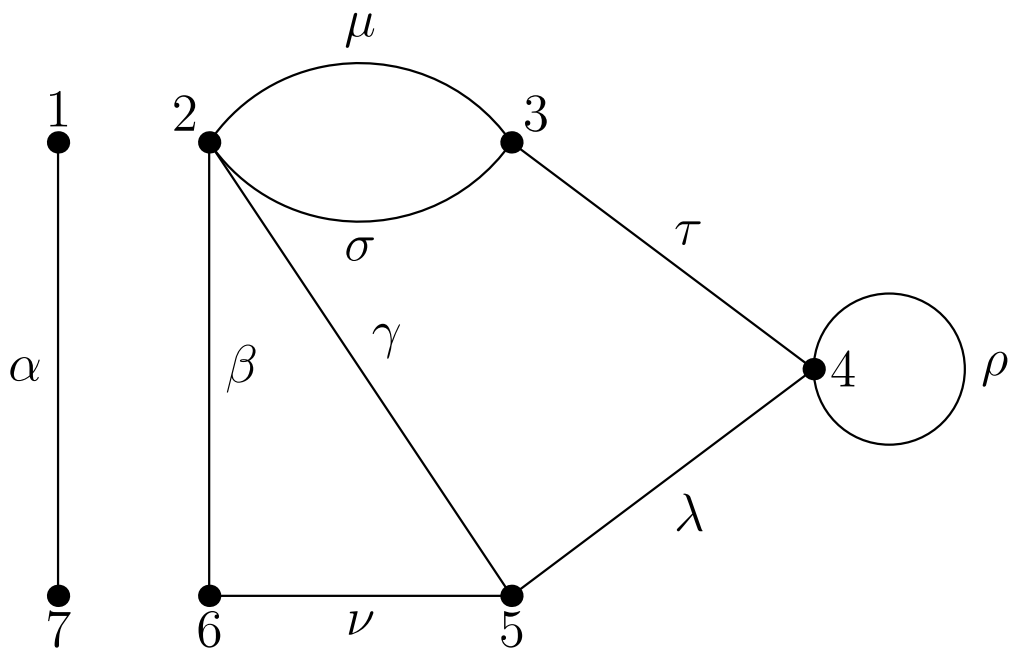
Définition

Soit s le sommet d'un graphe non-orienté. Le *degré* de s est égal au nombre d'arêtes dont s est une extrémité.

Attention, les boucles comptent double !

Exemple

$$d(1) = 1$$
$$d(2) = 4$$
$$d(4) = 4$$



Degré d'un sommet

Définition

Soit s le sommet d'un graphe orienté.

- le *degré sortant* $d^+(s)$ est le nombre d'arêtes dont s est le début
- le *degré entrant* $d^-(s)$ est le nombre d'arêtes dont s est la fin

Degré d'un sommet

Définition

Soit s le sommet d'un graphe orienté.

- le *degré sortant* $d^+(s)$ est le nombre d'arêtes dont s est le début
- le *degré entrant* $d^-(s)$ est le nombre d'arêtes dont s est la fin

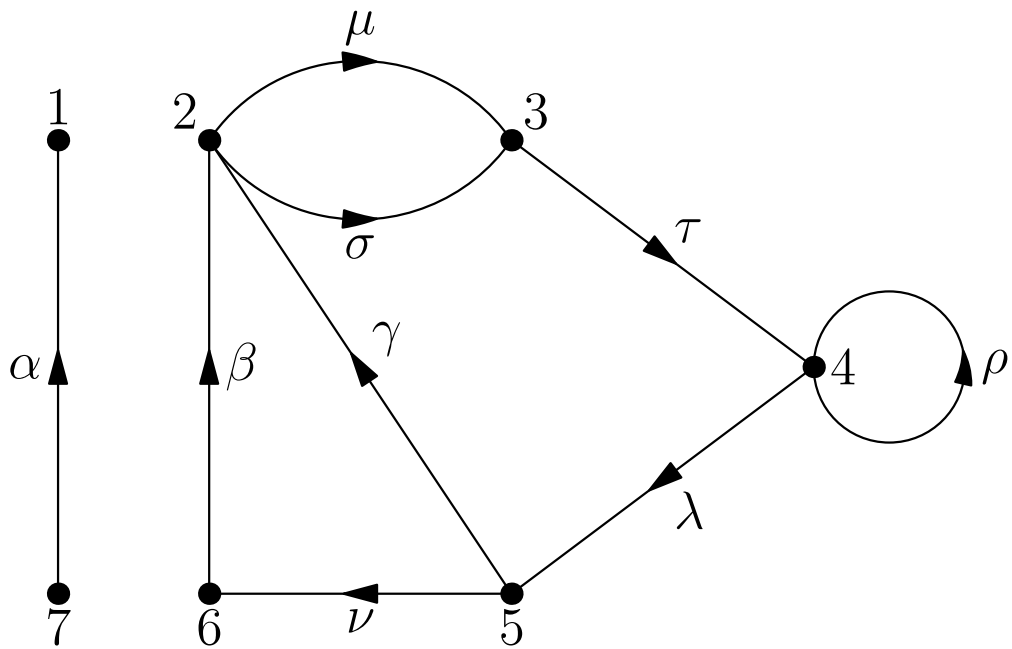
En effaçant le sens de parcours des arêtes, on obtient un graphe non-orienté. On a alors :

$$d(s) = d^+(s) + d^-(s)$$

Degré d'un sommet

Exemple

$$d^+(4) = 2$$
$$d^-(4) = 2$$
$$d(4) = 2 + 2 = 4$$



Chemin de longueur n

Définition

1. Soient s et t de sommets d'un graphe orienté $\mathscr{G} = (\mathcal{S}; \mathcal{A}; \delta)$. La suite $(S_0 = s, \varepsilon_1, S_1, \varepsilon_2, \cdots, \varepsilon_n, S_n = t)$ est un *chemin de longueur n* menant de s à t si :

$$\delta(\varepsilon_i) = S_{i-1}S_i$$

Chemin de longueur n

Définition

1. Soient s et t de sommets d'un graphe orienté $\mathscr{G} = (\mathcal{S}; \mathcal{A}; \delta)$. La suite $(S_0 = s, \varepsilon_1, S_1, \varepsilon_2, \cdots, \varepsilon_n, S_n = t)$ est un *chemin de longueur n* menant de s à t si :

$$\delta(\varepsilon_i) = S_{i-1}S_i$$

Autrement dit, on peut aller du sommet s au sommet t en suivant les arêtes du graphe et leur sens de parcours en n étapes.

Chemin de longueur n

Définition

1. Soient s et t de sommets d'un graphe orienté $\mathscr{G} = (\mathcal{S}; \mathcal{A}; \delta)$. La suite $(S_0 = s, \varepsilon_1, S_1, \varepsilon_2, \cdots, \varepsilon_n, S_n = t)$ est un *chemin de longueur n* menant de s à t si :

$$\delta(\varepsilon_i) = S_{i-1}S_i$$

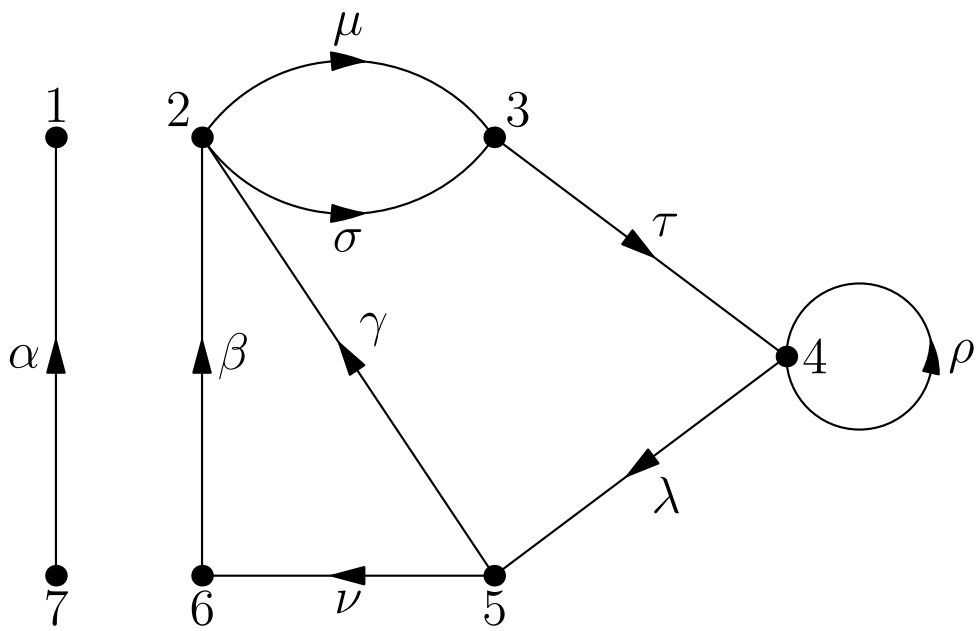
   Autrement dit, on peut aller du sommet s au sommet t en suivant les arêtes du graphe et leur sens de parcours en n étapes.

2. Un chemin dont le départ et l'arrivée coïncident est un *circuit*

Chemin de longueur *n*

Exemple

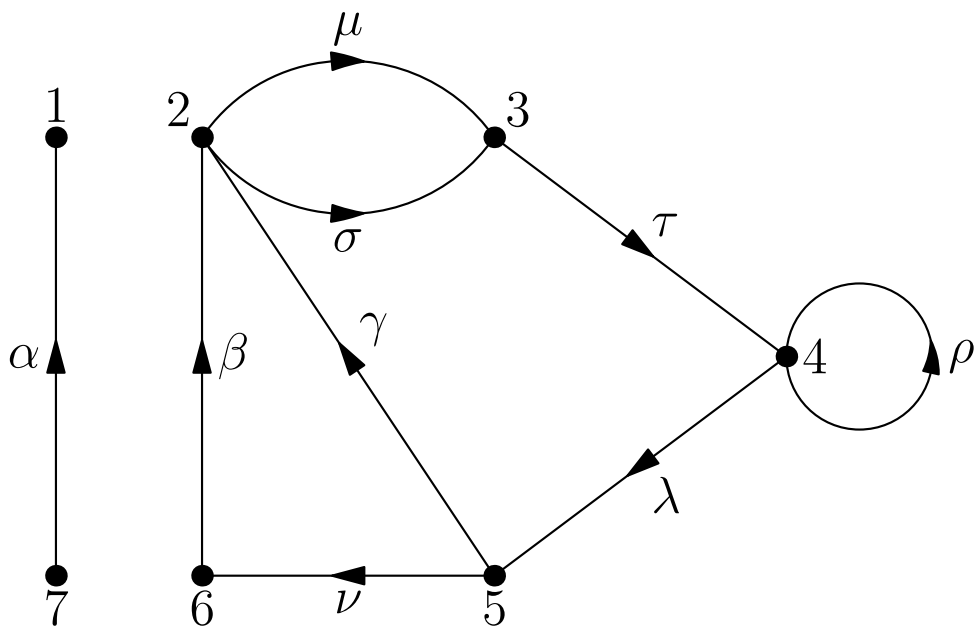
- $(5; \gamma; 2; \sigma; 3; \tau; 4)$ est un chemin de longueur 3 entre les sommets 5 et 4



Chemin de longueur *n*

Exemple

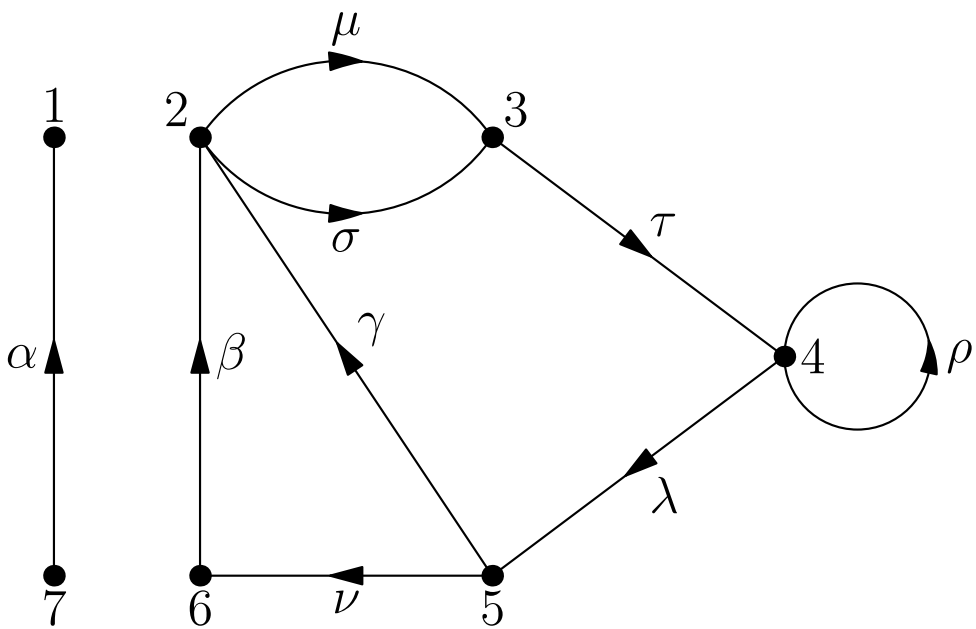
- $(5; \gamma; 2; \sigma; 3; \tau; 4)$ est un chemin de longueur 3 entre les sommets 5 et 4
- $(3; \tau; 4; \lambda; 5; \nu; 6; \beta; 2)$ est un chemin de longueur 4 entre les sommets 3 et 2



Chemin de longueur n

Exemple

- $(5; \gamma; 2; \sigma; 3; \tau; 4)$ est un chemin de longueur 3 entre les sommets 5 et 4
- $(3; \tau; 4; \lambda; 5; \nu; 6; \beta; 2)$ est un chemin de longueur 4 entre les sommets 3 et 2



Remarque

La définition est analogue pour les graphes non-orientés

Graphe connexe

Définition

1. Un graphe non-orienté est dit *connexe* s'il existe un chemin entre deux sommets quelconque de ce graphe
2. Un graphe orienté est dit *fortement connexe* s'il existe un chemin entre deux sommets quelconque de ce graphe

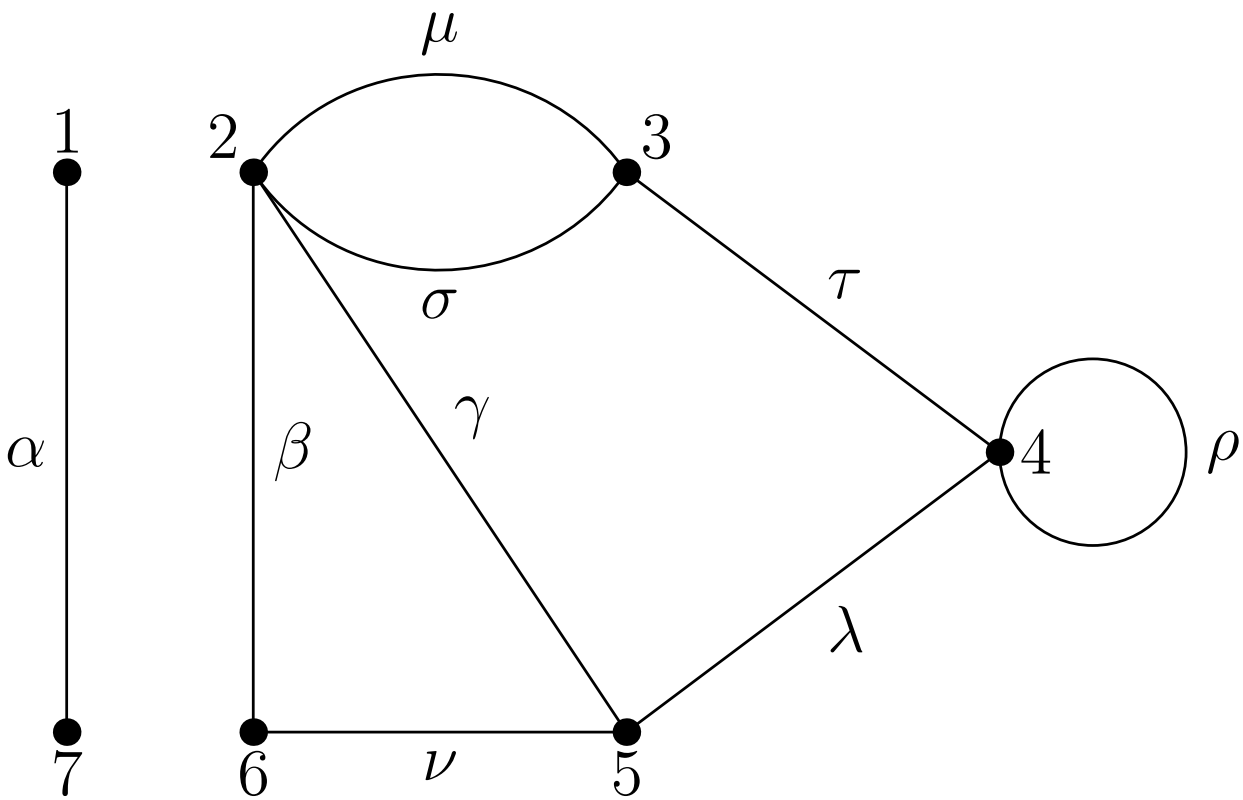
Graphe connexe

Définition

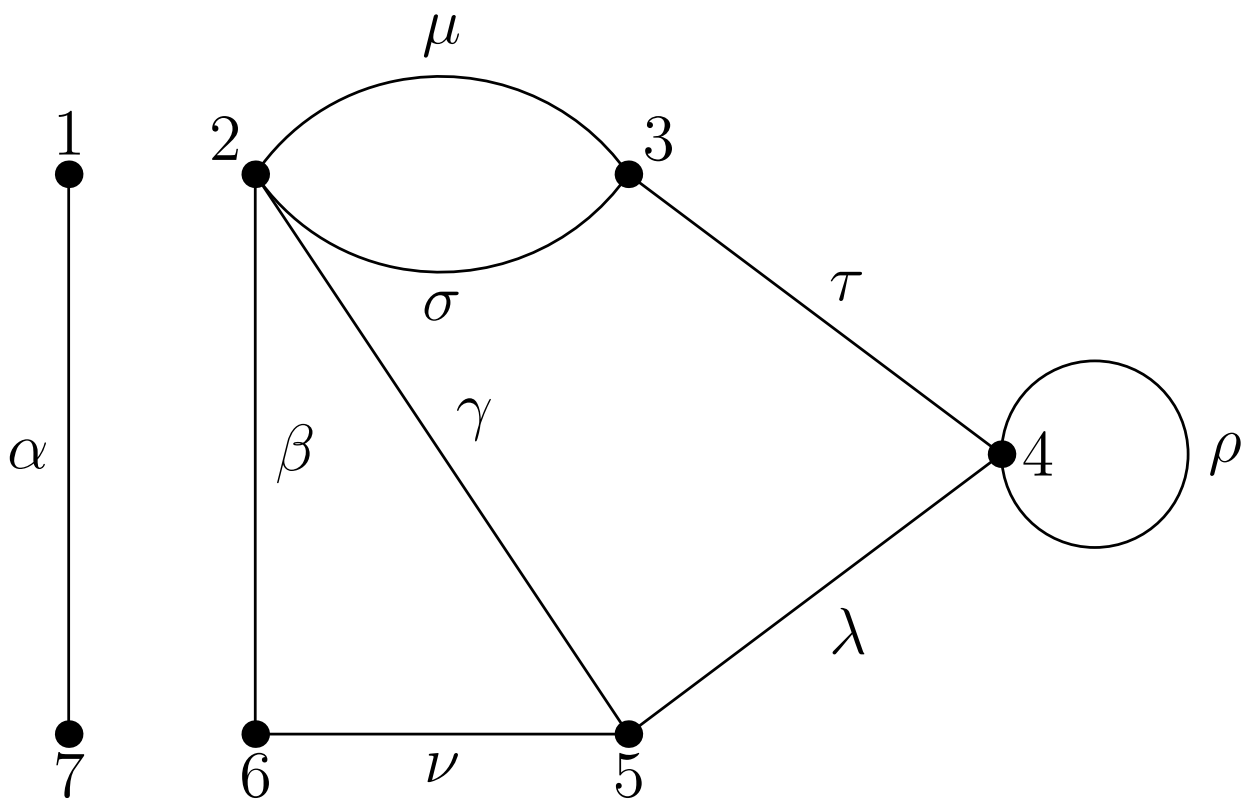
1. Un graphe non-orienté est dit *connexe* s'il existe un chemin entre deux sommets quelconque de ce graphe
2. Un graphe orienté est dit *fortement connexe* s'il existe un chemin entre deux sommets quelconque de ce graphe

Pour les graphes orientés, le sens de parcours des arêtes doit être respecté

Graphe connexe

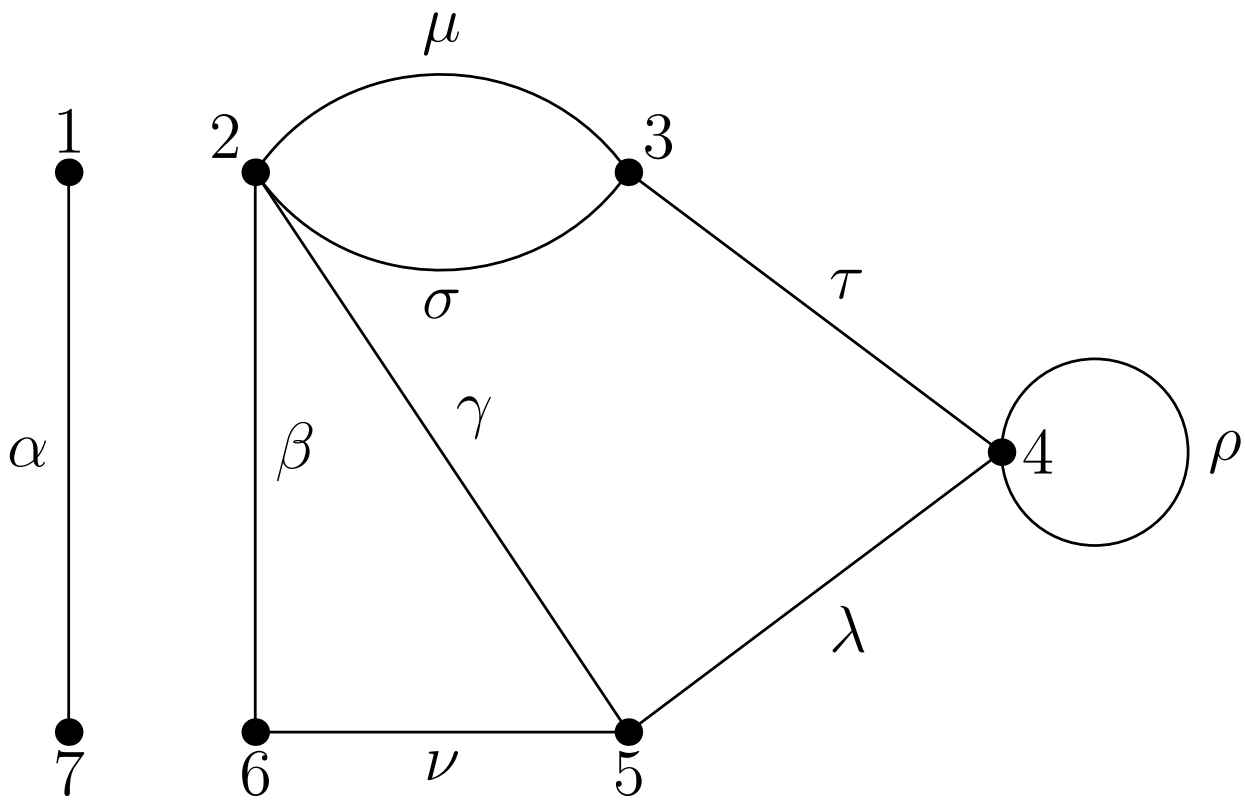


Graphe connexe



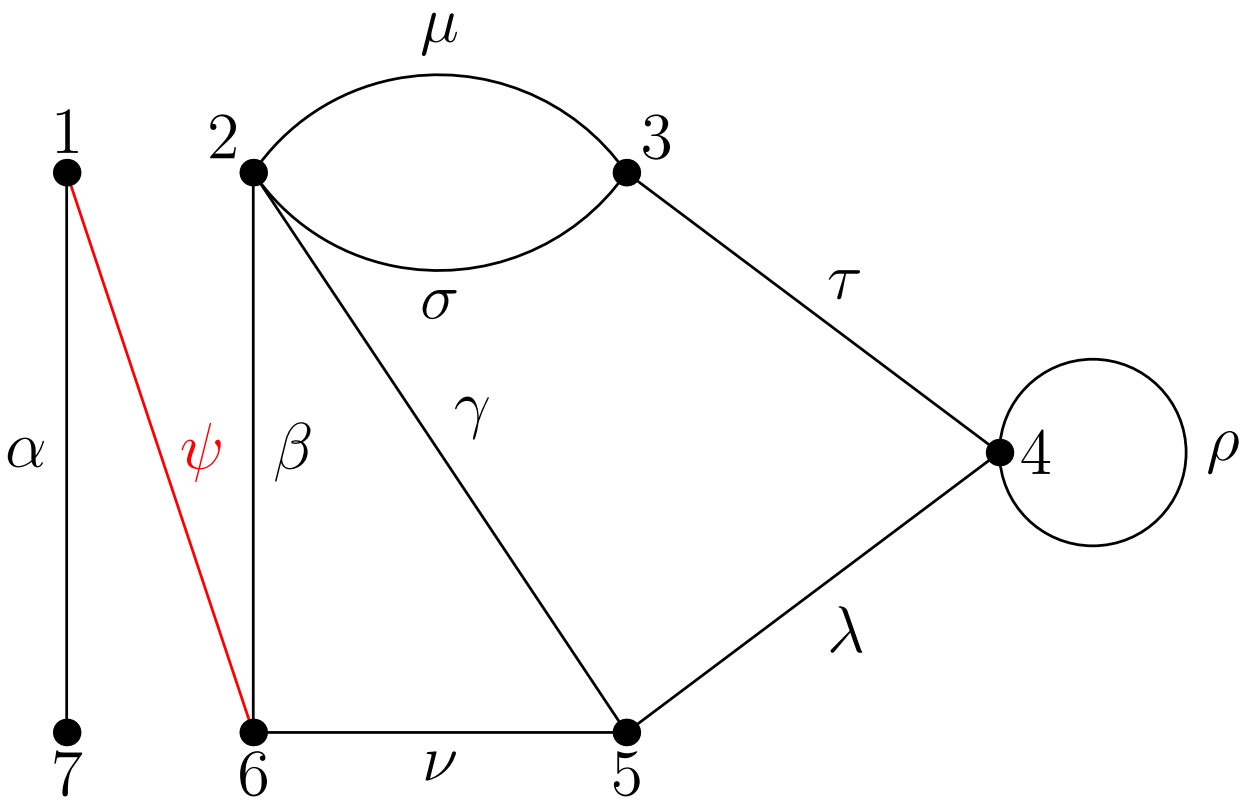
- $\mathscr{G}$ n'est pas connexe : on ne peut pas *passer* du sommet 1 au sommet 5

Graphe connexe



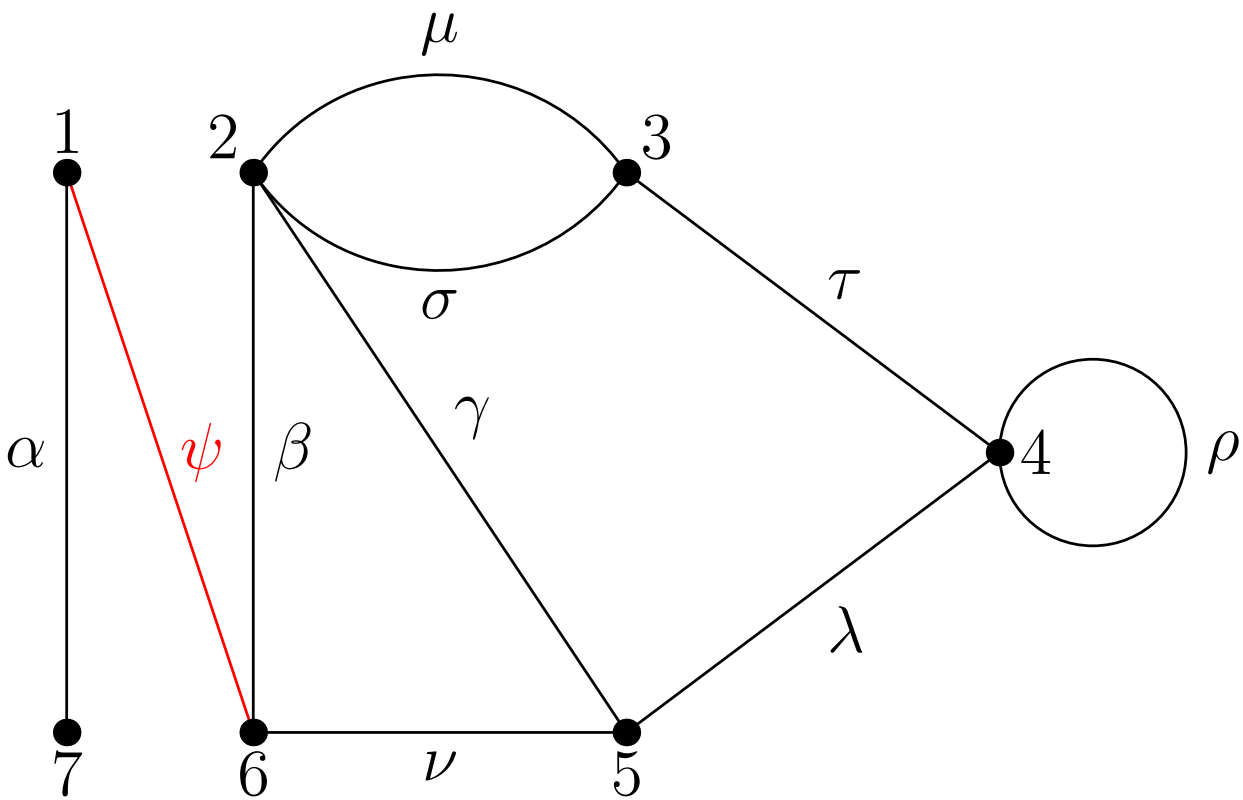
- $\mathscr{G}$ n'est pas connexe : on ne peut pas *passer* du sommet 1 au sommet 5
- On dit que $\mathscr{G}$ a deux *composantes connexes*

Graphe connexe



On ajoute l'arête $6 \to 1$

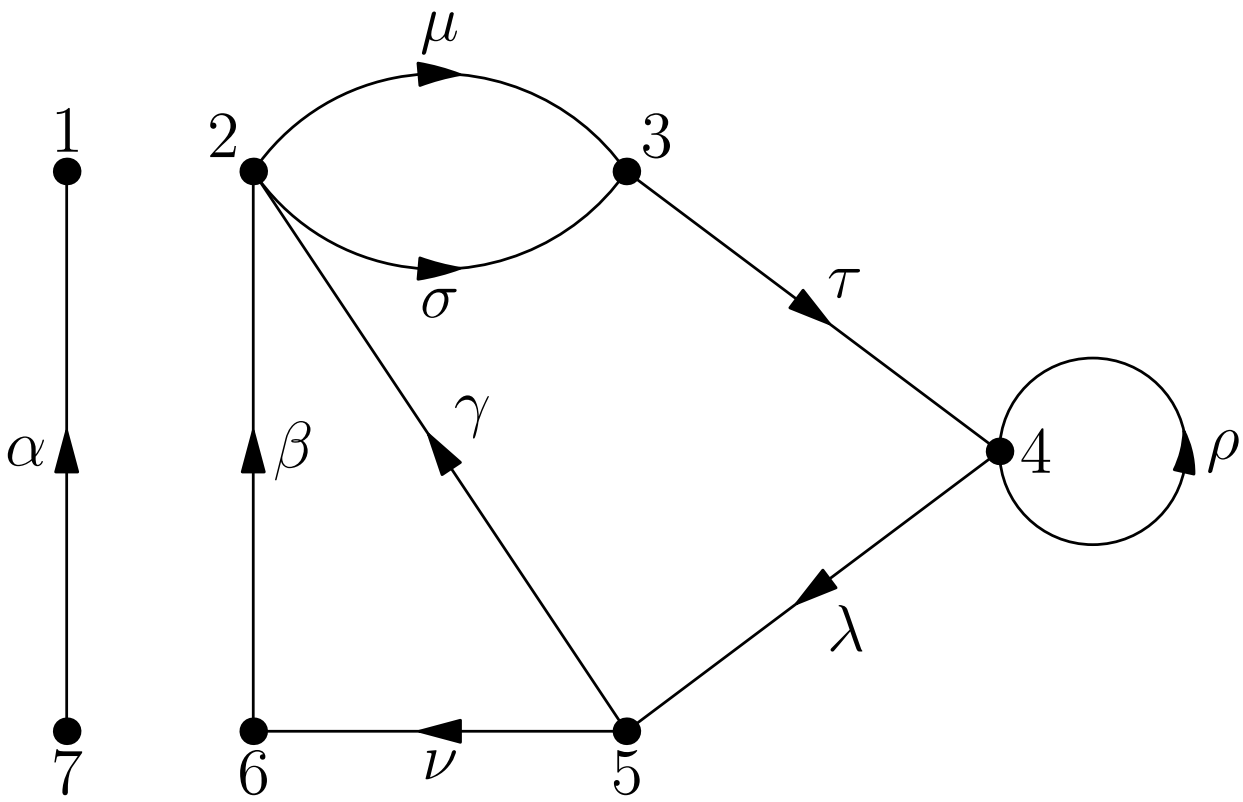
Graphe connexe



On ajoute l'arête $6 \to 1$

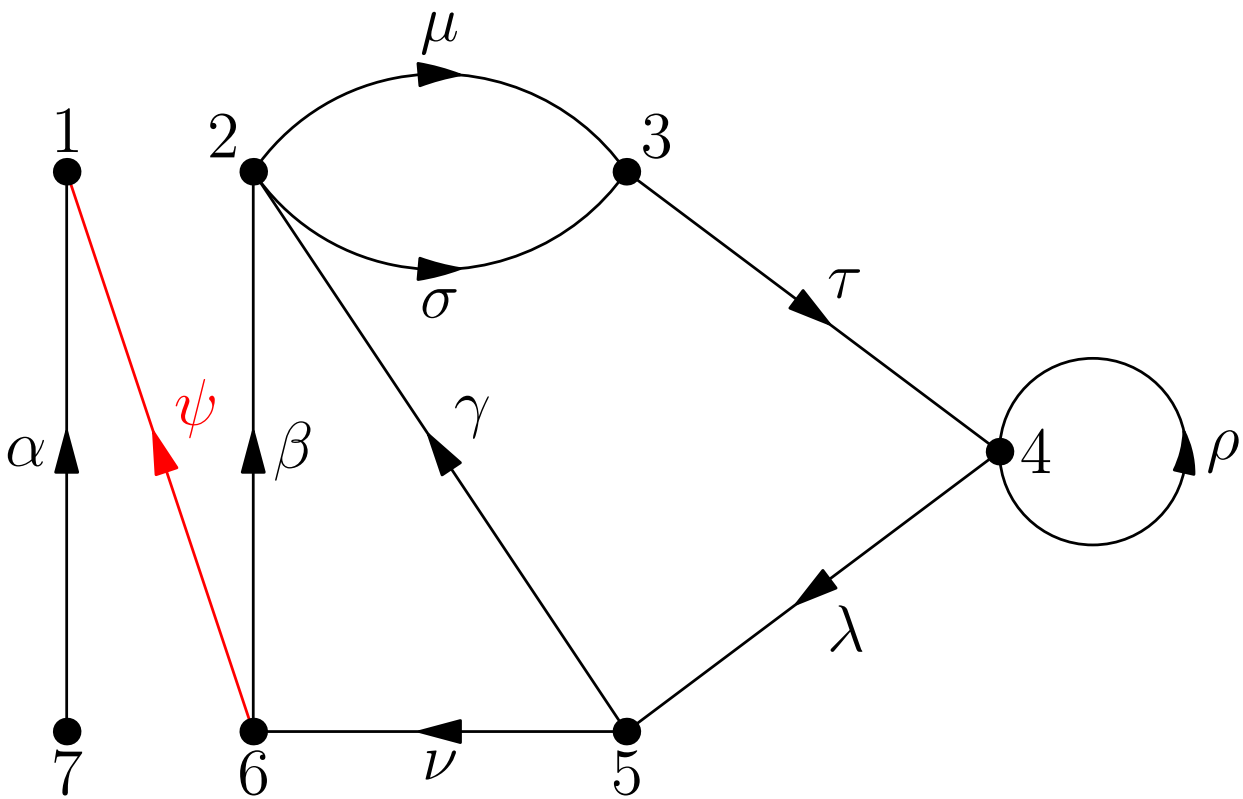
$\mathscr{G}$ est maintenant connexe

Graphe connexe



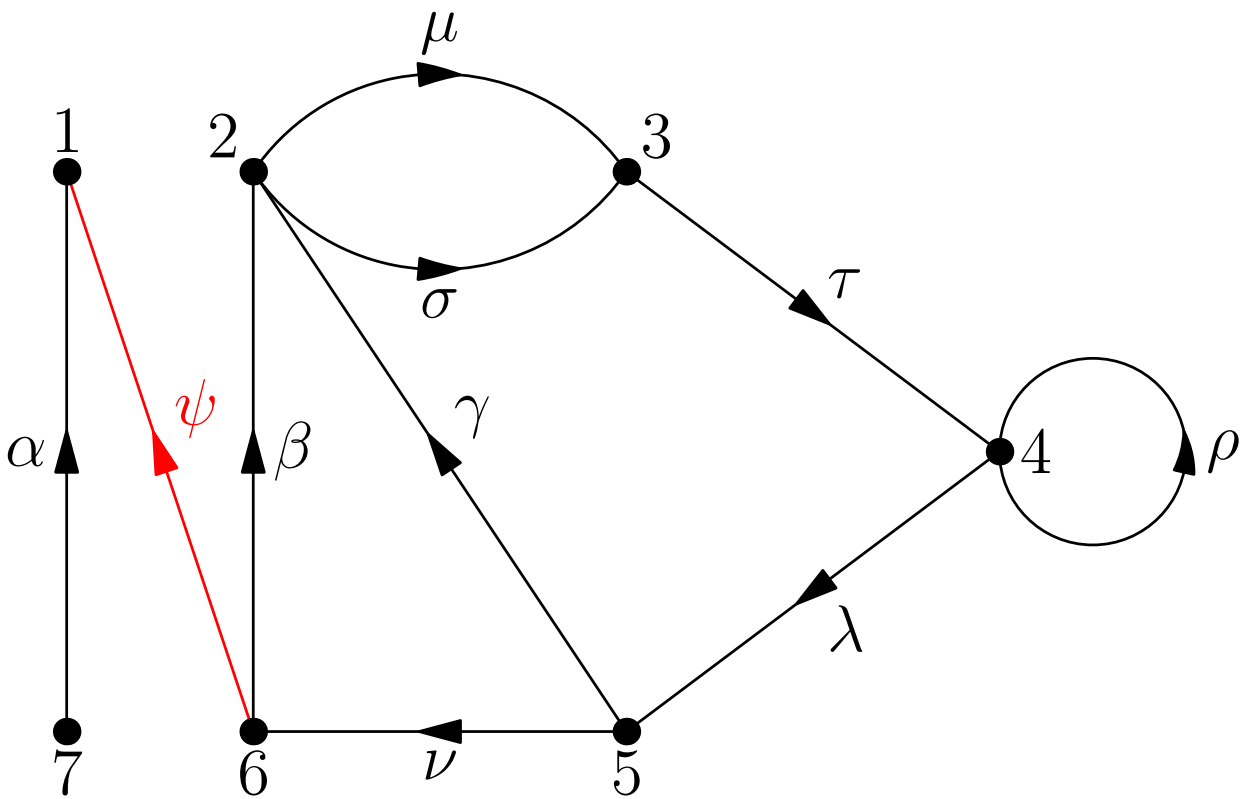
- $\mathscr{G}$ n'est pas fortement connexe : on ne peut pas *passer* du sommet 1 au sommet 5

Graphe connexe



On ajoute l'arête $6 \to 1$

Graphe connexe



On ajoute l'arête $6 \to 1$

$\mathscr{G}$ n'est pas fortement connexe : on ne peut (toujours) pas *passer* du sommet 1 au sommet 5

Matrice d'adjacence

Soit $\mathscr{G}$ un graphe orienté d'ordre n (n sommets). On suppose les sommets numérotés de 1 à n. On note S_1, S_2, ..., S_n ces sommets.

Définition

La matrice d'adjacence de $\mathscr{G}$ est la matrice $A \in \mathcal{M}_n(\mathbb{R})$ où :

$$a_{i,j} = \text{nombre d'arête de début } S_i \text{et de fin } S_j$$

Matrice d'adjacence

Soit $\mathscr{G}$ un graphe orienté d'ordre n (n sommets). On suppose les sommets numérotés de 1 à n. On note S_1, S_2, ..., S_n ces sommets.

Définition

La matrice d'adjacence de $\mathscr{G}$ est la matrice $A \in \mathcal{M}_n(\mathbb{R})$ où :

$$a_{i,j} = \text{nombre d'arête de début } S_i \text{et de fin } S_j$$

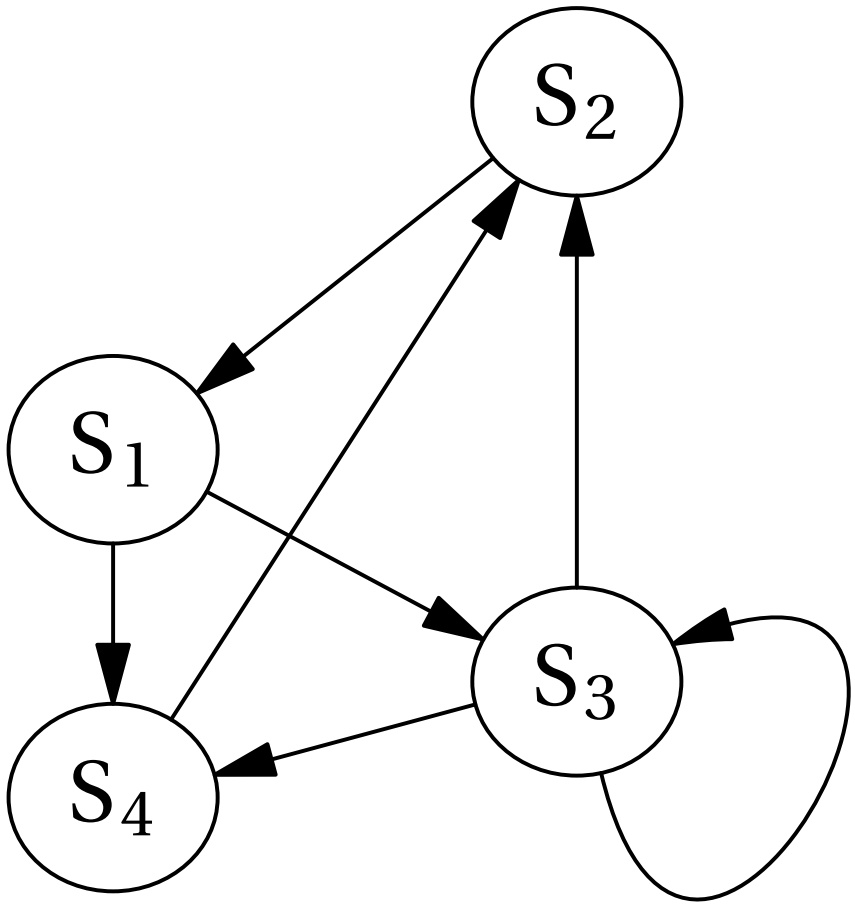
Remarque

La matrice A dépend de l'ordre dans le lequel on énumère les sommets

Matrice d'adjacence

Exemple

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



Matrice d'adjacence

Soit $\mathscr{G}$ un graphe non-orienté d'ordre n (n sommets). On suppose les sommets numérotés de 1 à n. On note S_1, S_2, ..., S_n ces sommets.

Définition

la matrice d'adjacence de $\mathscr{G}$ est la matrice $A \in \mathcal{M}_n(\mathbb{R})$ définie par :

$$a_{i,j} = \text{nombres d'arêtes entre les sommets } S_i \text{ et } S_j$$

Matrice d'adjacence

Soit $\mathscr{G}$ un graphe non-orienté d'ordre n (n sommets). On suppose les sommets numérotés de 1 à n. On note S_1, S_2, ..., S_n ces sommets.

Définition

la matrice d'adjacence de $\mathscr{G}$ est la matrice $A \in \mathcal{M}_n(\mathbb{R})$ définie par :

$$a_{i,j} = \text{nombres d'arêtes entre les sommets } S_i \text{ et } S_j$$

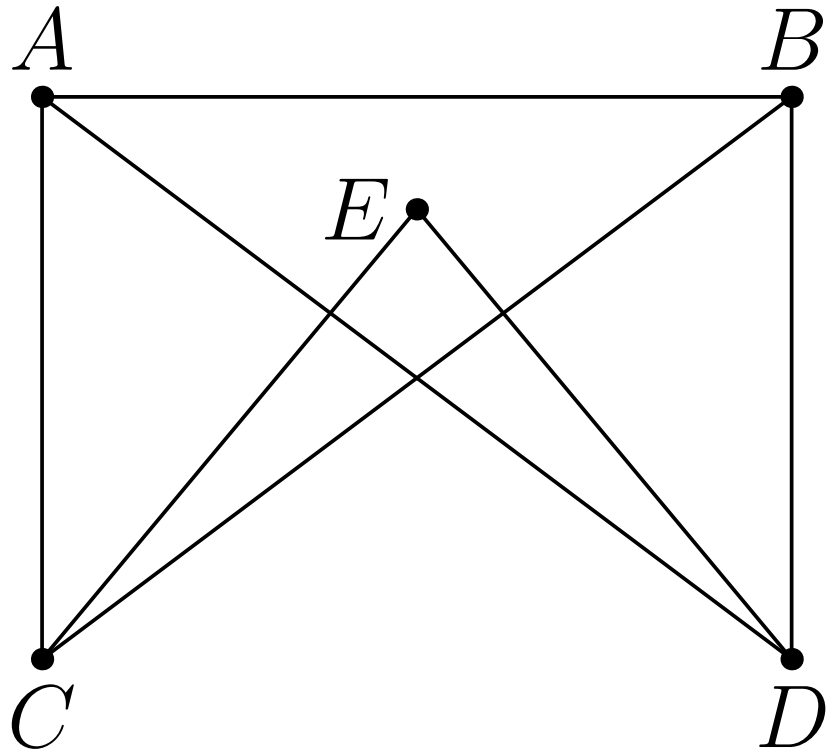
la matrice d'adjacence d'un graphe non-orienté est donc symétrique

Matrice d'adjacence

Exemple

En choisissant comme ordre des sommets l'ordre alphabétique :

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

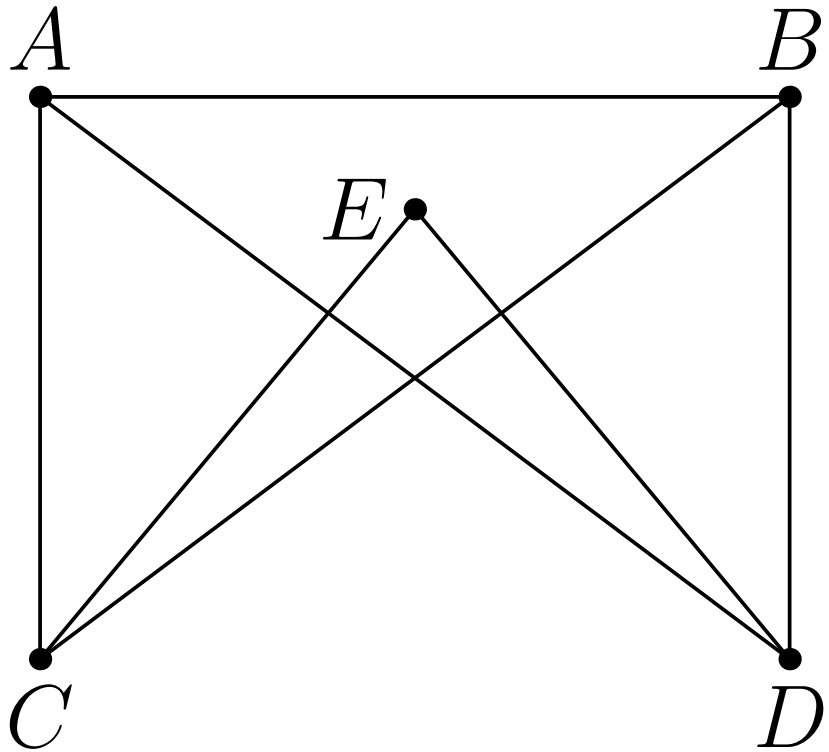


Matrice d'adjacence

Exemple

En choisissant comme ordre des sommets l'ordre alphabétique :

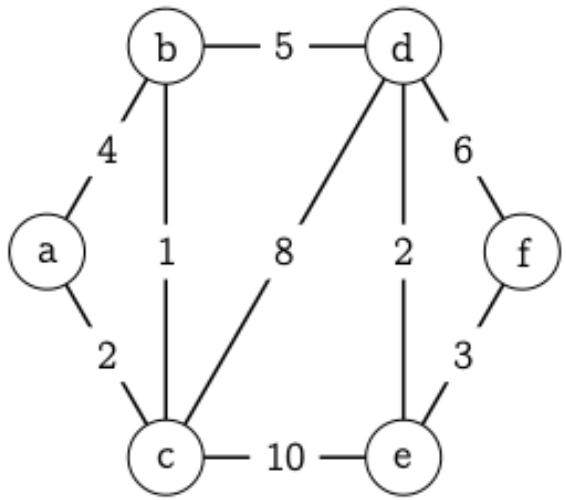
$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$



La matrice est bien symétrique

Graphes pondérés

C'est un graphe dont les arêtes sont affectées d'un « poids ».



Graphes pondérés

Définition

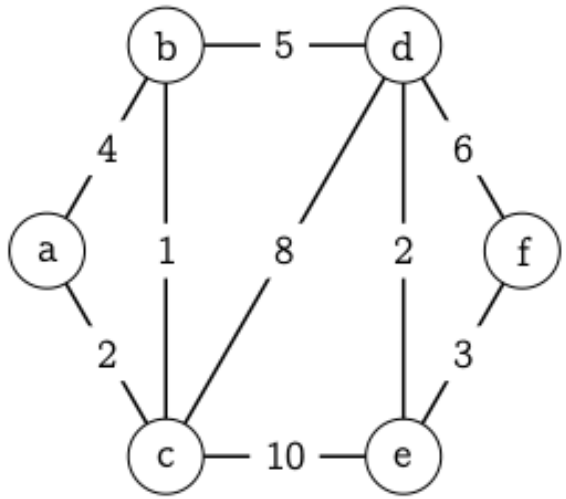
On appelle *matrice de pondération* d'un graphe $\mathscr{G}$ la matrice dont les coefficients correspondant aux sommets s et t valent :

$$A = \begin{cases} 0 & \text{si } t = s \\ \infty & \text{si } \{s,t\} \text{ n'est pas une arête} \\ p & \text{si } \{s,t\} \text{ est une arête de poids } p \end{cases}$$

Graphe pondéré

Exemple

$$\begin{pmatrix} 0 & 4 & 2 & \infty & \infty & \infty \\ 4 & 0 & 1 & 5 & \infty & \infty \\ 2 & 1 & 0 & 8 & 10 & \infty \\ \infty & 5 & 8 & 0 & 2 & 6 \\ \infty & \infty & 10 & 2 & 0 & 3 \\ \infty & \infty & \infty & 6 & 3 & 0 \end{pmatrix}$$

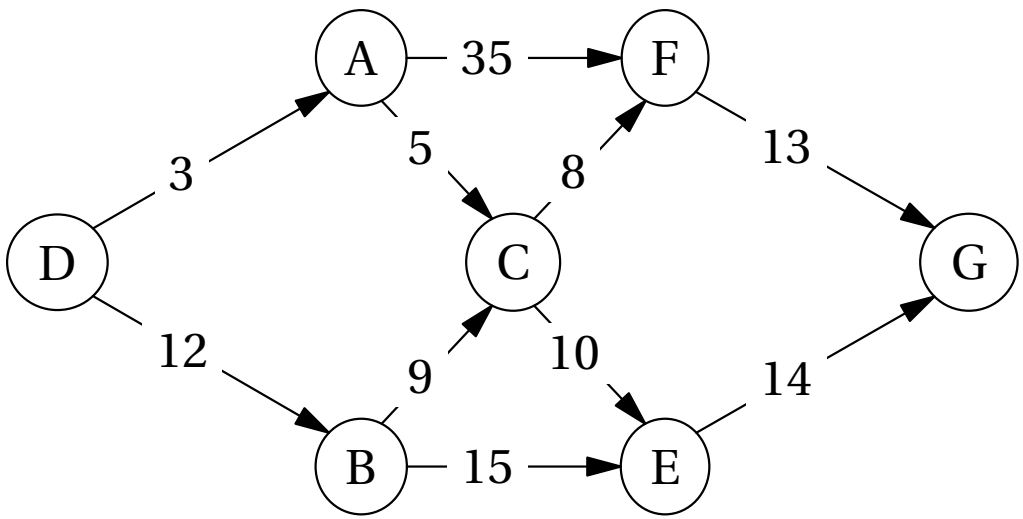


Sommaire

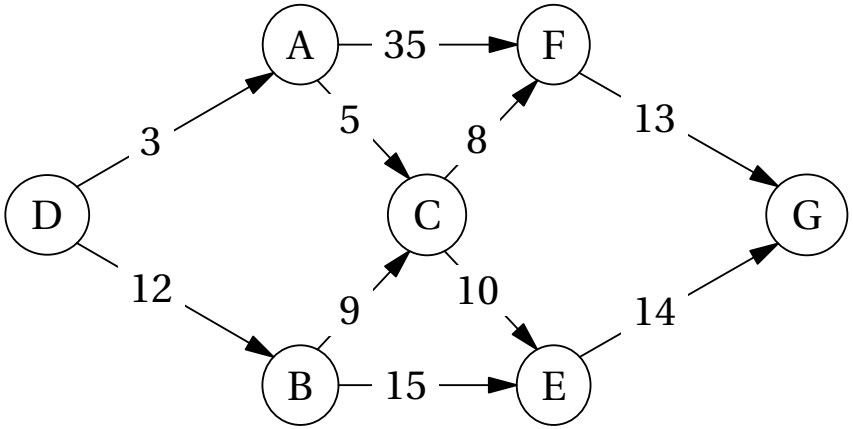
Algorithme de Dijkstra

L'algorithme de Dijkstra répond au problème du plus court chemin dans la cas où les poids sont positifs.

On cherche à déterminer la plus courte chaîne entre les sommets D et G.

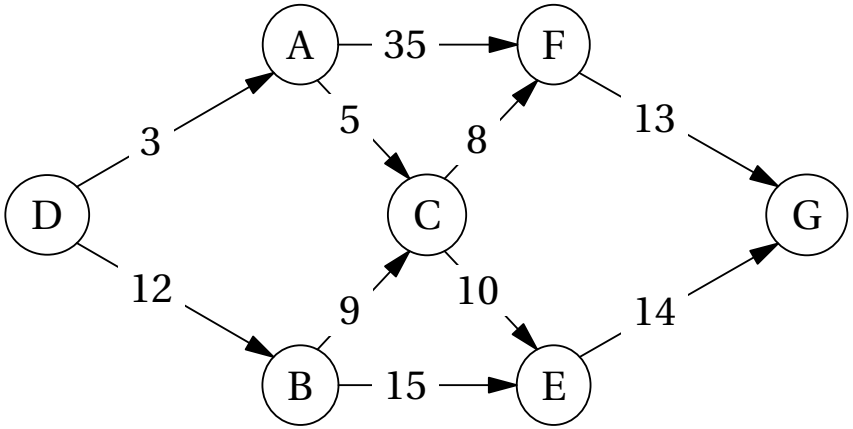


Algorithme de Dijkstra



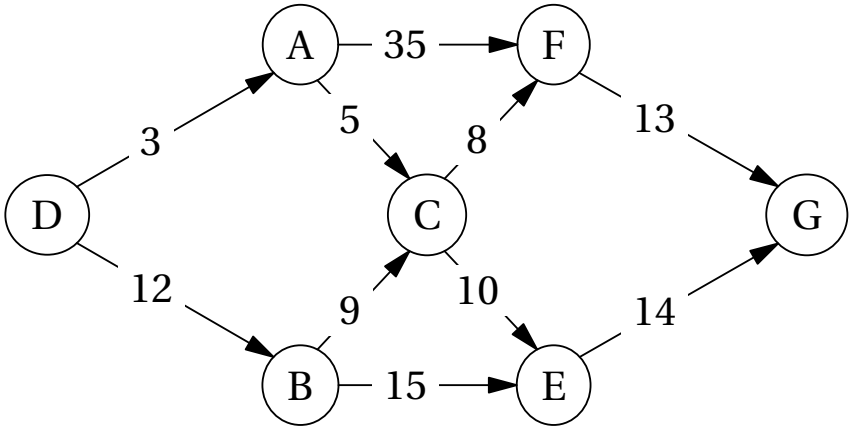
D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞

Algorithme de Dijkstra



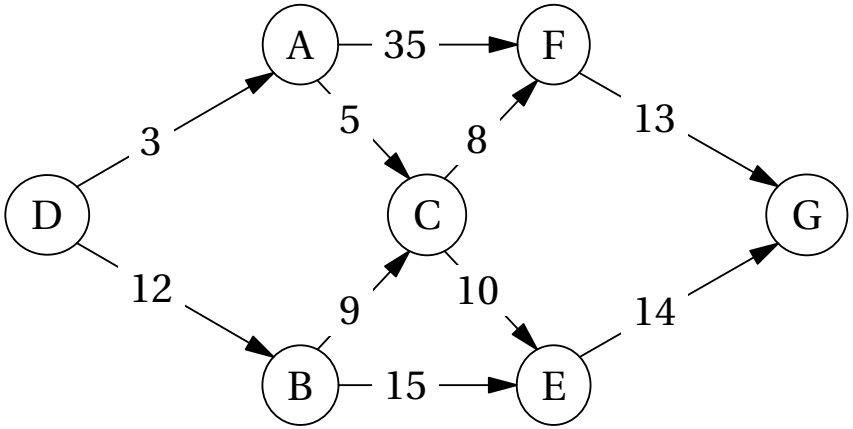
D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞
	3,D	12, D	8, A	∞	38, A	∞

Algorithme de Dijkstra



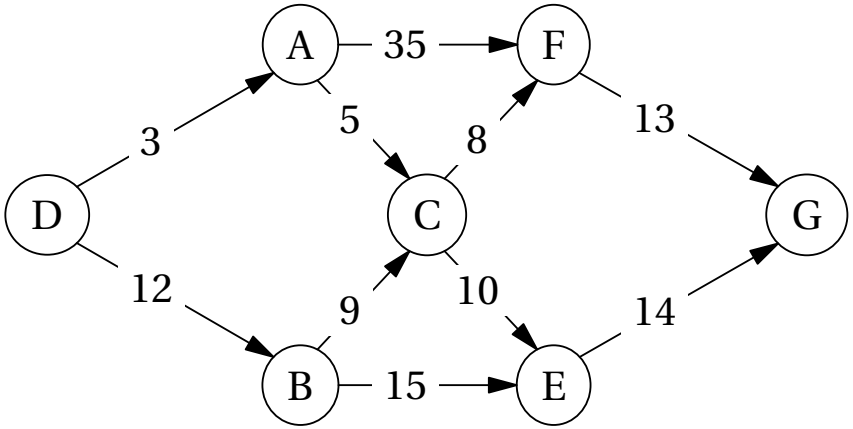
D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞
	3,D	12, D	8, A	∞	38, A	∞
		12, D	8, A	18, C	16, C	∞

Algorithme de Dijkstra



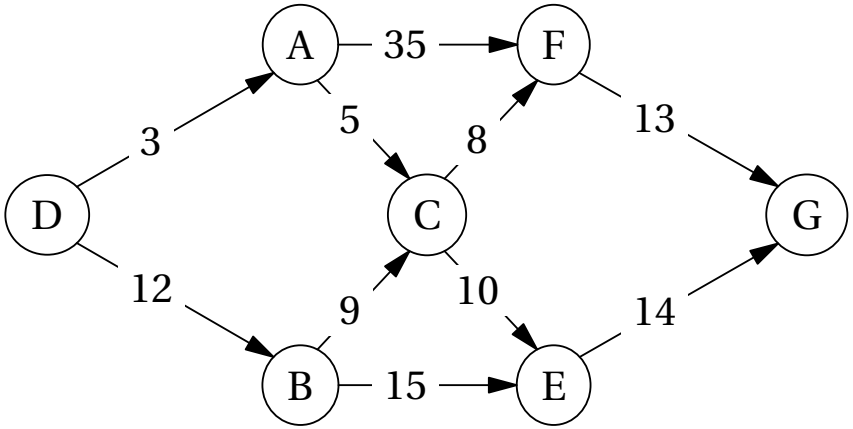
D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞
	3,D	12, D	8, A	∞	38, A	∞
		12, D	8, A	18, C	16, C	∞
		12, D		18, C	16, C	∞

Algorithme de Dijkstra



D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞
	3,D	12, D	8, A	∞	38, A	∞
		12, D	8, A	18, C	16, C	∞
		12, D		18, C	16, C	∞
				18, C	16, C	29, F

Algorithme de Dijkstra



D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞
	3,D	12, D	8, A	∞	38, A	∞
		12, D	8, A	18, C	16, C	∞
		12, D		18, C	16, C	∞
				18, C	16, C	29, F
				18,C		29, F

Algorithme de Dijkstra

D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞
	3,D	12, D	8, A	∞	38, A	∞
		12, D	8, A	18, C	16, C	∞
		12, D		18, C	16, C	∞
				18, C	16, C	29, F
				18,C		29, F

Algorithme de Dijkstra

D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞
	3,D	12, D	8, A	∞	38, A	∞
		12, D	8, A	18, C	16, C	∞
		12, D		18, C	16, C	∞
				18, C	16, C	29, F
				18,C		29, F

Pour lire la chaîne la plus courte, on part de la fin de parcours et on « remonte » la chaîne suivant les sommets de provenance

Algorithme de Dijkstra

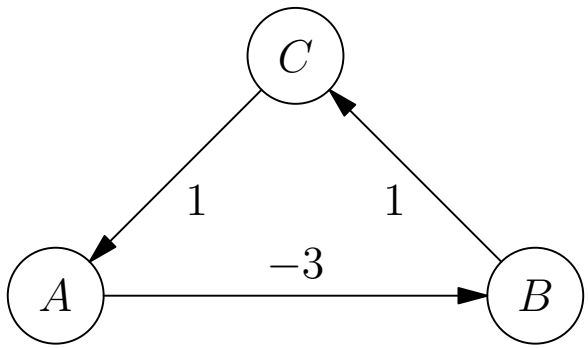
D	A	B	C	E	F	G
0	3, D	12, D	∞	∞	∞	∞
	3,D	12, D	8, A	∞	38, A	∞
		12, D	8, A	18, C	16, C	∞
		12, D		18, C	16, C	∞
				18, C	16, C	29, F
				18,C		29, F

Pour lire la chaîne la plus courte, on part de la fin de parcours et on « remonte » la chaîne suivant les sommets de provenance

$D - A - C - F - G$ de poids 29

Algorithme de Dijkstra : limites

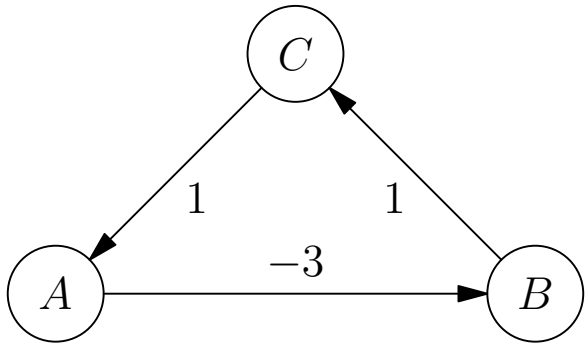
Et si les poids sont négatifs ?



- Le circuit $A \to B \to C \to A$ a un poids négatif : c'est un *circuit négatif*

Algorithme de Dijkstra : limites

Et si les poids sont négatifs ?



- Le circuit $A \to B \to C \to A$ a un poids négatif : c'est un *circuit négatif*

Le problème de la plus courte chaîne n'a pas de solution ici !

Il est possible de diminuer indéfiniment le poids d'une chaîne en utilisant ce circuit

Algorithme de Bellman-Ford

L 'algorithme de **Bellman-Ford** :

- recherche les plus courtes chaîne entre deux sommets . . .
- . . . mais qui permet l'utilisation de poids négatifs
- détecte les circuits négatifs

Principe :

- Le principe est le même que pour l'algorithme de Dijkstra
- mais les sommets ne sont plus marqués
  - il est possible de revenir sur certains sommets jusqu'à la fin de l'algorithme
  - avec des poids positifs, le poids total ne peut qu'augmenter. Ce n'est pas le cas ici

Algorithme de Bellman-Ford

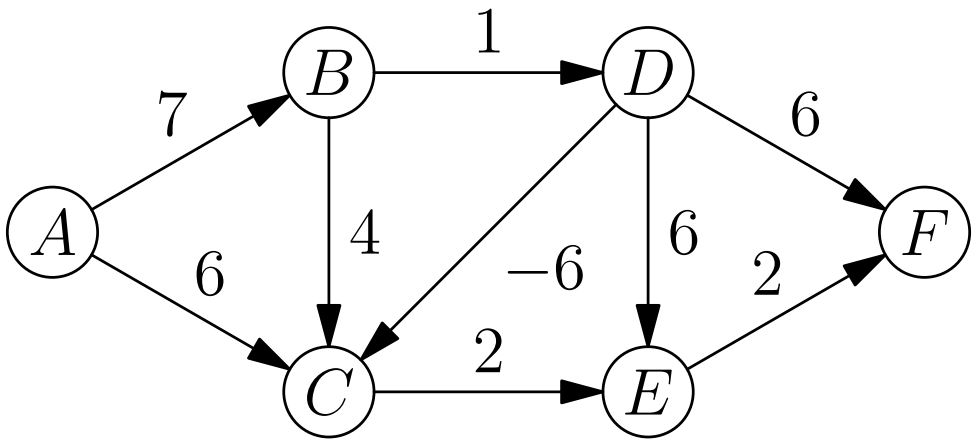
Arrêt :

1. Nombre d'itérations maximal : ordre du graphe
2. aucune valeur n'est modifiée entre deux itérations

Propriété

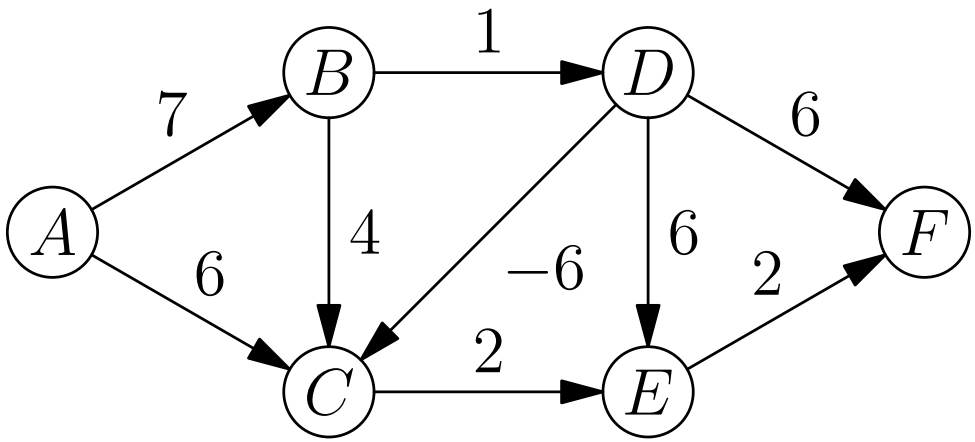
Si $\mathcal{G}$ est d'ordre n, et si les valeurs sont encore modifiées après n étapes, alors :
il existe un cycle négatif et il est inutile de continuer

Algorithme de Bellman-Ford



Au plus 6 étapes

Algorithme de Bellman-Ford

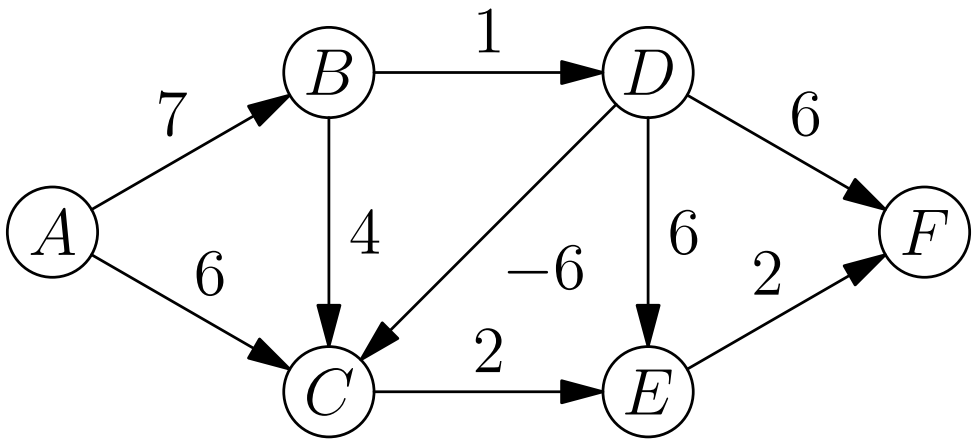


Au plus 6 étapes

La procédure d'initialisation est la même que dans l'algorithme de Dijkstra :

A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞

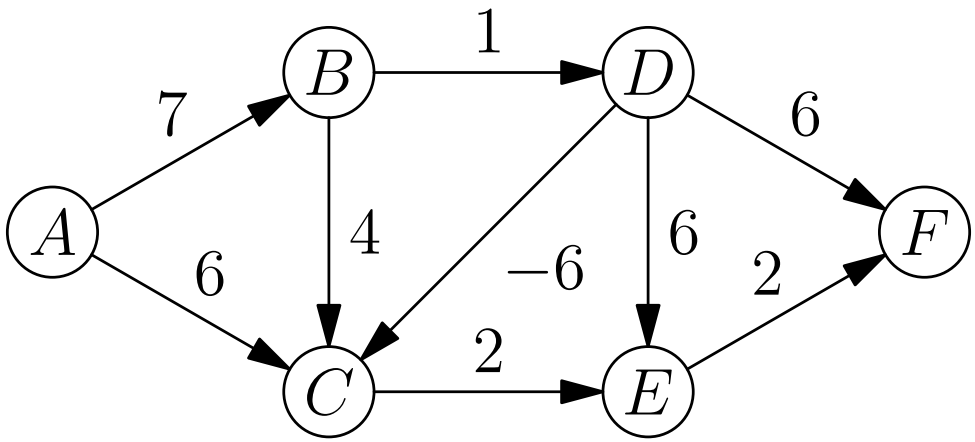
Algorithme de Bellman-Ford



Aucun sommet n'est marqué : on regarde où on peut aller à partir de A et de B :

A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞

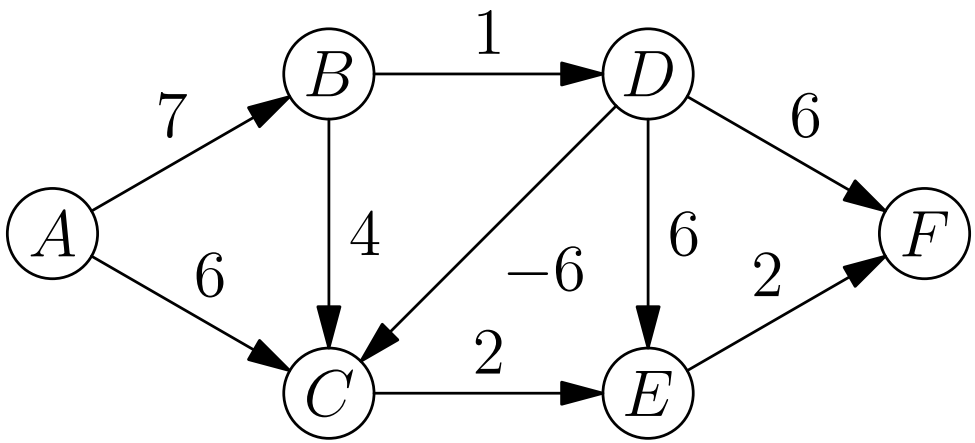
Algorithme de Bellman-Ford



Aucun sommet n'est marqué : on regarde où on peut aller à partir de A et de B :

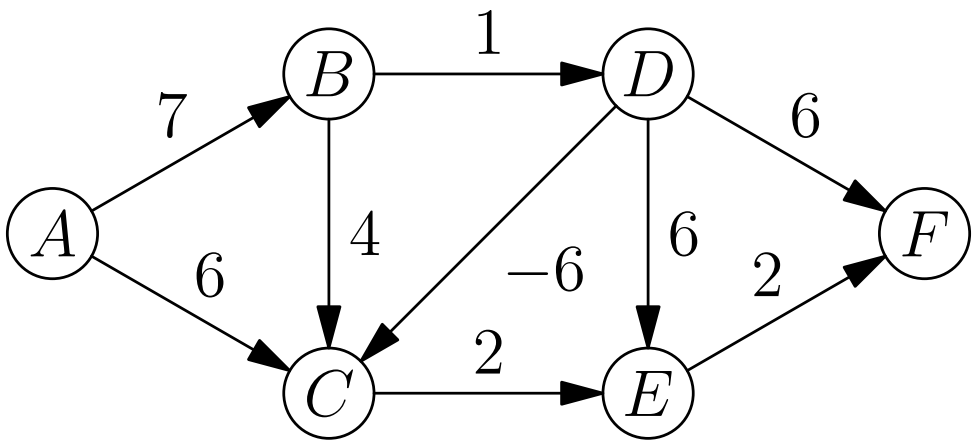
A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞
$(0, A)$	$(7, A)$	$(6, A)$	$(8, B)$	$(8, C)$	∞

Algorithme de Bellman-Ford



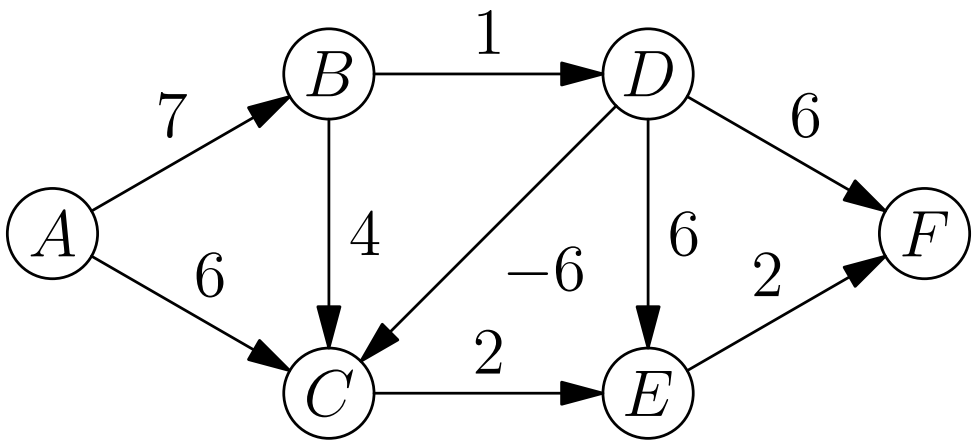
A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞
$(0, A)$	$(7, A)$	$(6, A)$	$(8, B)$	$(8, C)$	∞

Algorithme de Bellman-Ford



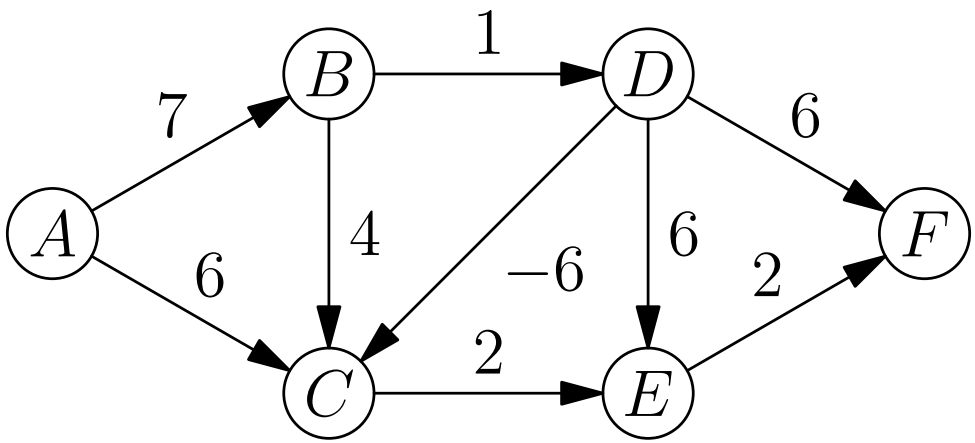
A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞
$(0, A)$	$(7, A)$	$(6, A)$	$(8, B)$	$(8, C)$	∞
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(8, C)$	$(10, E)$

Algorithme de Bellman-Ford



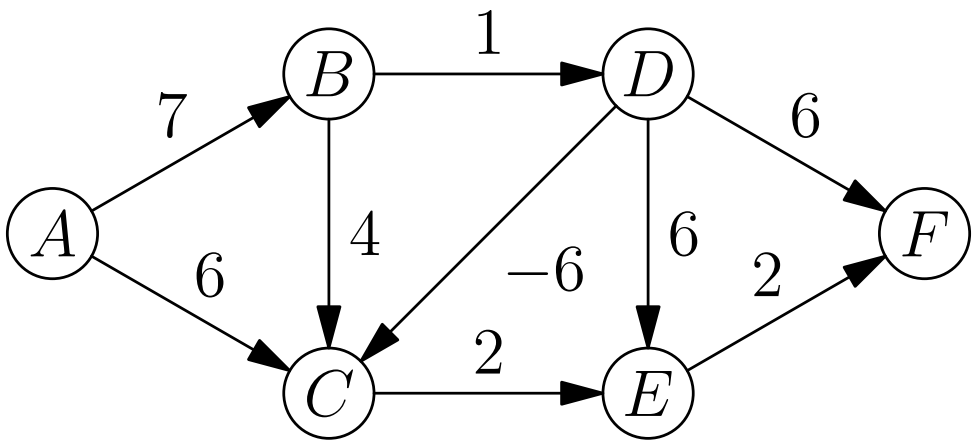
A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞
$(0, A)$	$(7, A)$	$(6, A)$	$(8, B)$	$(8, C)$	∞
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(8, C)$	$(10, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(10, E)$

Algorithme de Bellman-Ford



A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞
$(0, A)$	$(7, A)$	$(6, A)$	$(8, B)$	$(8, C)$	∞
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(8, C)$	$(10, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(10, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(6, E)$

Algorithme de Bellman-Ford



A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞
$(0, A)$	$(7, A)$	$(6, A)$	$(8, B)$	$(8, C)$	∞
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(8, C)$	$(10, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(10, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(6, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(6, E)$

Algorithme de Bellman-Ford

A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	∞	∞	∞
$(0, A)$	$(7, A)$	$(6, A)$	$(8, B)$	$(8, C)$	∞
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(8, C)$	$(10, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(10, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(6, E)$
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(6, E)$

- L'algorithme s'est terminé en 5 étapes
- il n'y a donc pas de circuit négatif

Algorithme de Bellman-Ford

Remarque :

- Avec Dijkstra, le sommet C aurait été marqué à la 1^{re} étape

Algorithme de Bellman-Ford

Remarque :

- Avec Dijkstra, le sommet C aurait été marqué à la 1^{re} étape
- On aurait obtenu le résultat (faux) :

A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(6, A)$	$(8, B)$	$(8, C)$	$(10, E)$

Algorithme de Bellman-Ford

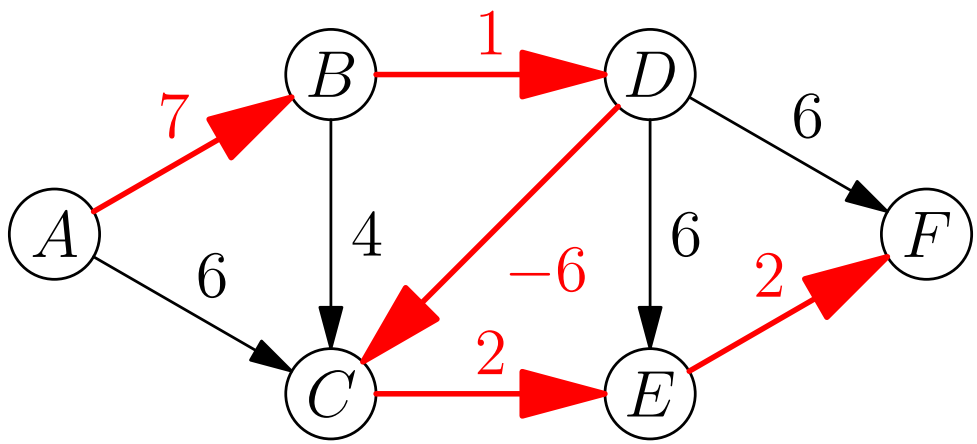
Le chemin le plus court est déduit comme dans l'algorithme de Dijkstra :

A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(6, E)$

Algorithme de Bellman-Ford

Le chemin le plus court est déduit comme dans l'algorithme de Dijkstra :

A	B	C	D	E	F
$(0, A)$	$(7, A)$	$(2, D)$	$(8, B)$	$(4, C)$	$(6, E)$



Sommaire

Décomposition en niveau

Les graphes orientés **sans circuit** possèdent des propriétés spécifiques : en particulier il existe dans un graphe sans circuit une notion de hiérarchie entre les sommets. C'est ce qu'on appelle la **décomposition en niveaux** ou aussi un « tri topologique »

Théorème (Décomposition en niveaux d'un graphe sans circuit)

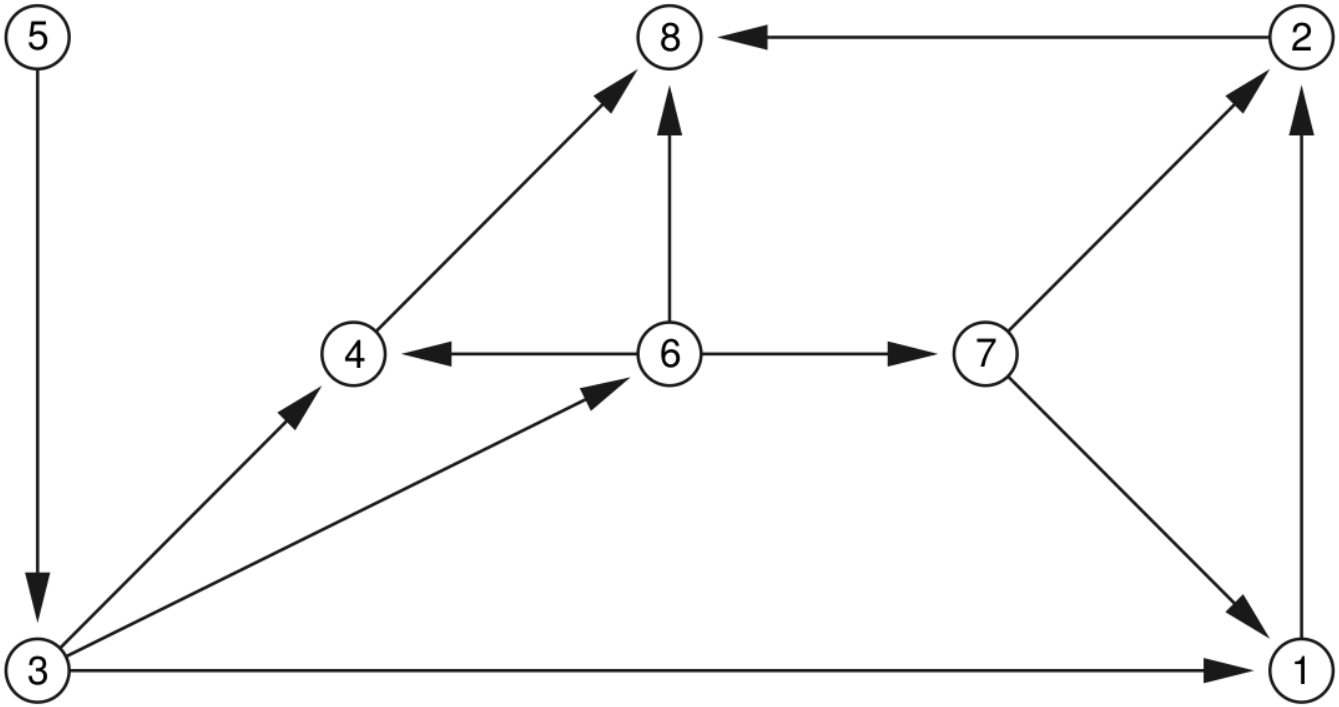
Si G est un graphe sans circuit, alors on peut définir pour chaque sommet un niveau de la manière suivante :

- *les sommets sans prédécesseurs sont de rang 0 ;*
- *tout sommet x a un rang supérieur aux rangs de ses prédécesseurs :*

$$rang(x) = \max_{y \in N^-(x)} rang(y) + 1.$$

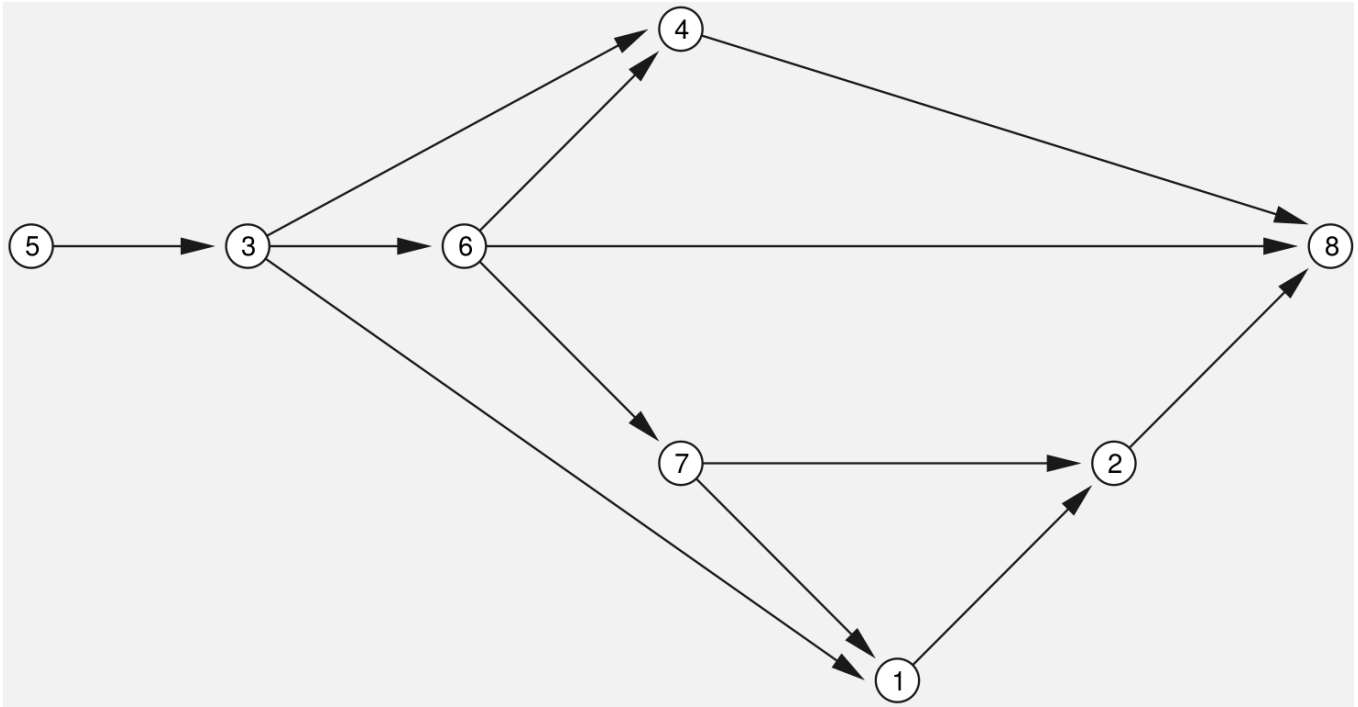
Décomposition en niveau

Considérons le graphe orienté suivant :



Décomposition en niveau

On peut redessiner le graphe en utilisant la numérotation topologique :

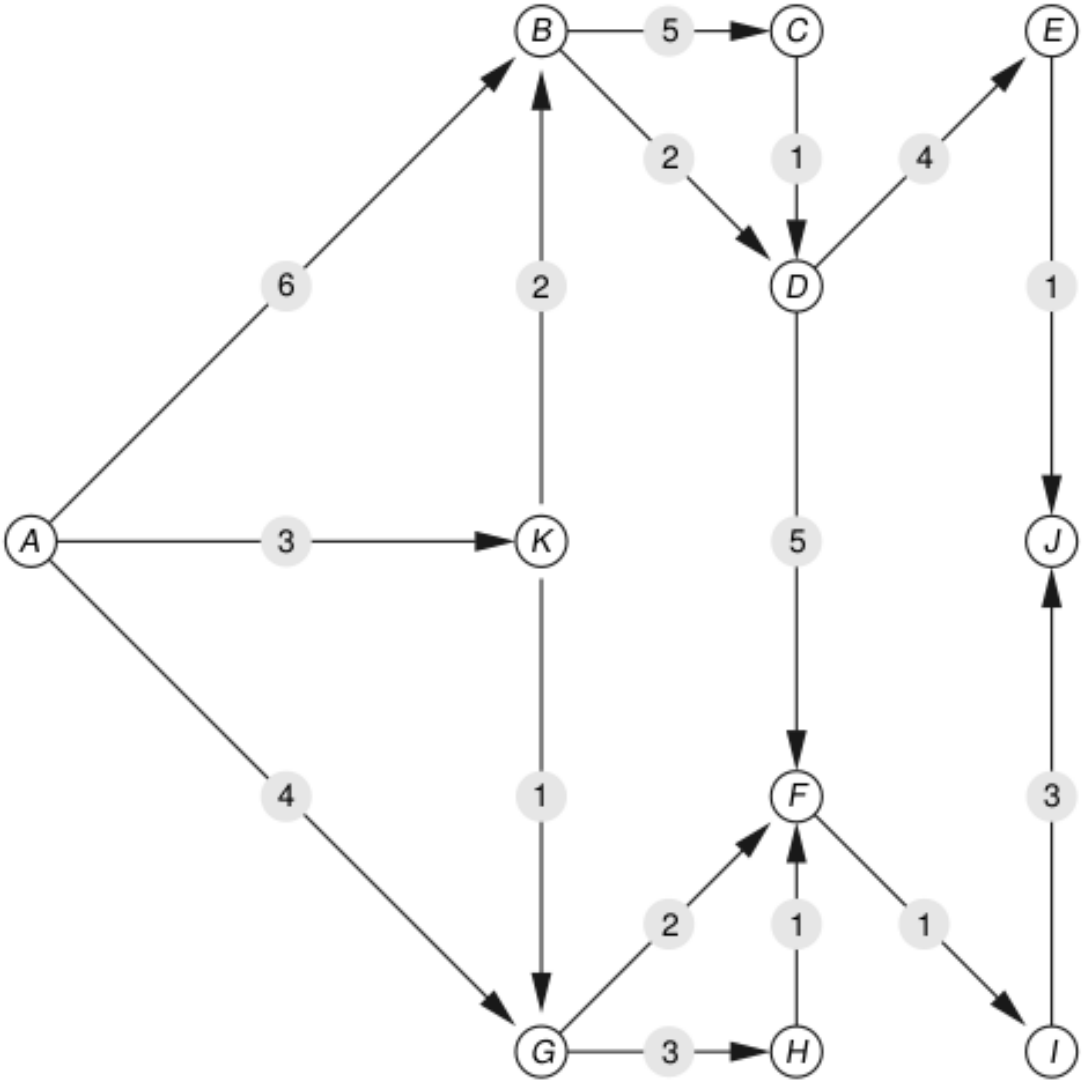


Algorithme de Bellman

Variables : $d(i)$ distance de 1 à i ;
$P(i)$ prédecesseur de i
Initialisation : $d(1) = 0$;
$P(1) = \emptyset$;
pour $i = 2, \cdots, n$ **faire**
    $d(i) = +\infty$;
    $P(i) = \emptyset$;
fin

1 **début**
2 **pour** $i = 2, \cdots, n$ **faire**
3 $d(i) = \min_{jN^-(i)} (d(j) + f(i,j))$;
4 $j_0 =$ prédécesseur de i fournissant la valeur de $d(i)$ ci-desssous ;
5 $P(i) = j_0$;
6 **fin**
7 **fin**

Algorithme de Bellman



Sommaire