Chap. 2 Plus courts chemins

On présente dans ce chapitre un problème typique de cheminement dans les graphes : **la recherche d'un plus court chemin entre deux sommets**.

Ce problème a des nombreuses applications :

- Trouver le moyen le plus économique pour aller de Brest à Lyon, connaissant pour chaque ligne aérienne le prix de billet d'avion.
- Les problèmes d'optimisation de réseaux (réseaux routiers ou réseaux de télécommunications
- Les problèmes d'ordonnancement
- Les problèmes d'intelligence artificielle tels que la circulation dans un labyrinthe

3.1 Définition, conditions d'existence et exemples

- G = [S, A] est **valué** ⇔ à chaque arc **u** est associé une longueur (un coût) **l(u)**.

- On appelle coût cumulé d'un chemin μ (ou tout simplement coût ou poids de μ) noté coût(μ) ou l(μ), la somme des coûts des arcs qui le compose. On appelle plus court chemin de x vers y un chemin allant de x à y de coût minimum. Un tel chemin n'existe pas toujours.

$$l(\mu) = \sum_{u \in \mu(x,y)} l(u)$$

- La longueur l(u) peut s'interpréter différemment dans de nombreuses applications pratiques.

  - Carte routière et chemin de moindre coût
  - Construction d'une autoroute entre 2 villes

Conditions d'existence :

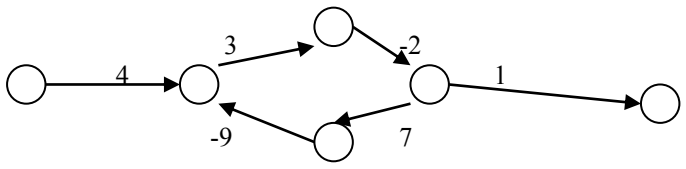
Considérons un chemin μ allant de x à y; supposons ce chemin contient un circuit Γ. Soit μ' le chemin obtenu en supprimant ce circuit de μ. On a :

coût(μ) = coût(μ') + coût(Γ)

Si coût(Γ) est négatif, il n'existe pas de plus court chemin de x à y, car étant donné un chemin λ de x à y, on peut toujours construire un chemin λ' de coût strictement inférieur à λ, en passant une fois de plus dans Γ.

Si coût(Γ) est positif ou nul, μ' est de coût inférieur ou égal à μ; c'est un meilleur candidat que μ pour être solution du problème si coût(Γ) est strictement positif.

Les circuits de coût négatif sont appelés **circuits absorbants**.



Dans la suite on suppose qu'il n'existe pas de tels circuits. Un plus court chemin ne peut contenir un circuit de coût positif. S'il y a des circuits de coût nul, il y a plusieurs solutions au problème de la recherche d'un plus court chemin.

Comme le nombre de chemins élémentaires de x vers y (chemins qui ne contiennent pas plusieurs fois un même sommet) est fini, il existe au moins un plus court chemin élémentaire de x vers y.

Un plus court chemin élémentaire est un plus court chemin de x vers y, s'il n'y a pas de circuit absorbant.

Le problème de la recherche d'un plus court chemin de x vers y, admet une solution dès qu'il existe un chemin de x vers y, et qu'il n'y a pas de circuits absorbants.

3.2 Variantes de problème

Le problème de la recherche d'un plus court chemin se rencontre sous l'une des trois formes suivantes :

1. on recherche un plus court chemin entre deux sommets donnés

2. on recherche les plus courts chemins entre un sommet donné, appelé source et tous les autres sommets

3. on recherche les plus courts chemins entre tous les sommets pris deux à deux.

On ne connaît pas de solution meilleure pour le problème 1 que celle qui consiste à passer par les solutions de 2.

Il existe différents algorithmes qui sont plus ou moins bien adaptés selon les propriétés du graphe et la forme du problème étudié (2 ou 3), mais il n'y a pas d'algorithme général qui soit efficace et intéressant dans tous les cas.

3.3 Algorithme de Dijkstra

Cet algorithme n'est utilisable que dans le cas, très fréquent, où les coûts des arcs sont tous positifs ou nuls. Il calcule un plus court chemin entre **source s et tous les sommets accessibles depuis s** : on obtient alors une arborescence de racine s formée par ces plus courts chemins.

Principe

- On considère dans la présentation suivante que sommet 1 est un sommet source

- Soit G = [S, A] ; S = {1, 2, …, N}. On initialise tous les $\pi(i) = l_{1i}$. l_{ij} = la longueur de l'arc (i,j) si $(i,j) \in A$, sinon $l_{ij} = \infty$. $l_{ii} = 0$.

- On note $\pi^*(i)$ la longueur minimum des chemins de 1 à i ; donc $\pi^*(1) = 0$.

- L'ensemble des sommets S est partitionné en 2 : M et CC. M = S – CC. Au début CC={1} et M = {2, 3, …, N}. Donc, on construit progressivement un ensemble CC des sommets x pour lesquels on connaît le plus court chemin de 1 vers x.

Tant que M n'est pas vide

  - On choisit dans l'ensemble M un sommet j tel que $\pi(j)$ est le plus petit et on ajoute le sommet j dans l'ensemble CC.

  - Pour tous les sommets i qui sont successeurs de j on met dans $\pi(i)$ le minimum de l'ancien $\pi(i)$ et la somme de $\pi(j)$ et l'arc qui mène de j à i.

Algorithme

a) Initialisation

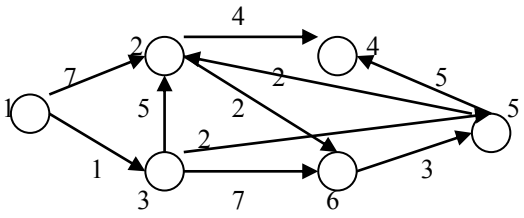
CC={1} ; M = {2, 3, …, N}

$\pi(1)=0$; $\pi(i) = \begin{cases} l_{1i} \text{ si } i \in \Gamma_1 \\ +\infty \text{ sinon} \end{cases}$

b) TANT QUE $|M| \neq 0$ FAIRE

Sélectionner $j \in M$ tel que $\pi(j)=\min_{i \in M}\pi(i)$
FAIRE $M=M - \{j\}$; $CC = CC + \{j\}$
SI $|M| \neq 0$ ALORS
POUR TOUT $i \in \Gamma_j$ et $i \in M$ FAIRE
$\pi(i) \leftarrow \min(\pi(i), \pi(j) + l_{ji})$

Ex.



Init.
CC = {1} ; M = {2, 3, 4, 5, 6}
$\pi(2)=7$; $\pi(3) = 1$, $\pi(4) =+\infty$, $\pi(5) =+\infty$, $\pi(6) =+\infty$

Sélect. 3.
M = {2, 4, 5, 6} ; CC = {1, 3}
$\pi(2) = \min(7, 1+5)=6$
$\pi(5) = \min(+\infty, 1+2)=3$
$\pi(6) = \min(+\infty, 1+7)=8$

Sélect. 5
M = {2, 4, 6} ; CC = {1, 3, 5}
$\pi(2) = \min(6, 3+2)=5$
$\pi(4) = \min(+\infty, 3+5)=8$

Sélect. 2
M = {4, 6} ; CC = {1, 3, 5, 2}
$\pi(4) = \min(8, 5+4)=8$
$\pi(6) = \min(8, 5+2)=7$

Sélect. 6
M = {4} ; CC = {1, 3, 5, 2, 6}

Sélect. 4
M = {∅} ; CC = {1, 3, 5, 2, 6, 4}
Résultat

$\pi^*(1)=0$

$\pi^*(3)=1$

$\pi^*(5)=3$

$\pi^*(2)=5$

$\pi^*(6)=7$

$\pi^*(4)=8$

3.4 Cas où l(u) = 1, ∀u∈A

C'est le cas particulier où toutes les longueurs sont **égales**. On calcule la longueur du plus court chemin d'un sommet (sommet 1, par exemple) à **tous les autres sommets**.

Principe.

- On définit $\pi(i)$ comme étant le plus court chemin du sommet 1 au sommet i
- On initialise $\pi(i) = +\infty$, $\forall i \geq 2$. Dès que $\pi(i)$ devient $\neq +\infty$, elle indique la longueur du plus court chemin.
- Le parcours du graphe est un parcours en largeur et chaque sommet n'est visité qu'une **seule fois**.
- La valeur de $\pi(i)$ correspond au nombre **d'arcs minimum** pour atteindre i.

Algorithme

a) Poser $\pi(1) = 0$; $\pi(i) = +\infty$; $\forall i \geq 2$, $i \in S$; k=0; CC={1} ;
$S_0 = \{1\}$.

b) **TANT QUE** $|CC| \neq |S|$ et $\Gamma(S_k) \neq \varnothing$ **FAIRE**

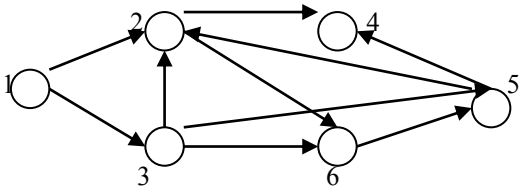
/* A l'itération k, soit $S_k = \{i \mid \pi(i) = k, i \in S\}$ et
$CC = \{i \mid \pi(i) \leq k, i \in S\}$ */

Poser $S_{k+1} \leftarrow \Gamma(S_k) \cap (S \backslash CC)$

$\pi(i) \leftarrow k+1, \forall i \in S_{k+1}$

$CC \leftarrow CC \cup S_{k+1}$

SI $|CC| \neq |S|$ **ALORS** $k \leftarrow k + 1$



Résultat du parcours d'exemple.

k=0
$S_0 = \{1\}$
CC = {1}
$S_1 = \{2, 3\}$
$\pi(2) = 1$
$\pi(3) = 1$
CC = {1, 2, 3}

k=1
$S_1 = \{2, 3\}$
CC = {1, 2, 3}
$S_2 = \{4, 5, 6\}$
$\pi(4) = 2$
$\pi(5) = 2$
$\pi(6) = 2$
CC = {1, 2, 3, 4, 5, 6}

3.5 Graphe G et longueurs l(u) quelconques : algorithme de Bellman

Cet algorithme est valable pour des graphes sans circuits négatifs, valués par des coûts quelconques.

Il donne un plus court chemin d'un sommet source à tous les autres sommets.

- Soit $\pi(i)$ la longueur du plus court chemin de 1 à i, et l'arc (i, j) est de longueur l_{ij}.
- Pour le plus court chemin de 1 à j pouvant passer par i, on doit avoir :

$$\pi(j) \leq \pi(i) + l_{ij} \quad \Rightarrow \quad \pi(j) - \pi(i) \leq l_{ij}$$

- π peut donc être considéré comme un potentiel : la différence de potentiel entre i et j doit être inférieure ou égale à l_{ij}.

Principe

- Soit G=[S, A] un graphe dont les arcs sont munis de longueurs réelles quelconques.
- Lorsque la procédure se termine, $\pi(i)$ représente la valeur du plus court chemin entre 1 et i.
- Les $\pi(i)$, $\forall i \in S$ sont modifiés itérativement de façon à satisfaire à la condition d'optimilité suivante :

  Un ensemble de valeurs $\pi^*(i)$, i = 1, 2, …, N et $\pi^*(1) = 0$ représente les longueurs des plus courts chemins de 1 aux autres sommets de G $\Leftrightarrow$ ces valeurs vérifient l'ensemble des inéquations suivantes :

  $$\forall (i, j) \in A \ \pi^*(j) \leq \pi^*(i) + l_{ij}. \ (****)$$

- L'idée : parcourir séquentiellement la liste de tous les arcs et vérifier la condition (****)

- Le problème est résolu si la condition (****) est vérifiée pour tous les arcs (i, j).

Algorithme

a) Poser $\pi^0(1)$ et $\pi^0(i) = +\infty$, $\forall i \neq 1$; k = 1.

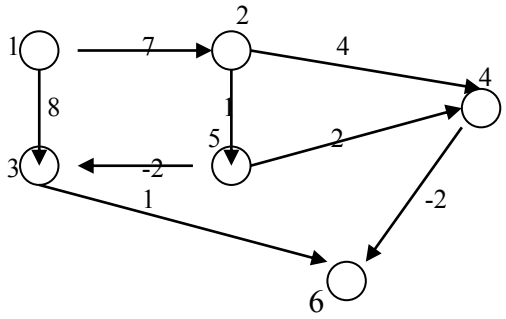
b) A l'itération k faire $\pi^k(i) = \min(\pi^{k-1}(i), \min_{j \in \Gamma_i^{-1}}(\pi^{k-1}(j) + l_{ji}))$

c) SI $\exists$ i | $\pi^k(i) \neq \pi^{k-1}(i)$ ALORS

SI $k \leq N - 1$ ALORS $k \leftarrow k + 1$
SINON /* k = N */ Il existe un circuit de longueur négative.

- Donc, si le graphe n'admet pas de circuit de longueur négative, les valeurs finales sont obtenues en au plus N – 1 itérations.

  Autrement dit, si au bout de N itérations les $\pi(i)$ continuent à être encore modifiés alors le graphe comporte un circuit de longueur négative.



Exemple.

k = 1
$\pi^1(2) = \min(7, +\infty) = 7$
$\pi^1(3) = \min(8, +\infty) = 8$
Les autres sont toujours à $+\infty$.

k = 2
$\pi^2(2) = 7$
$\pi^2(3) = 8$
$\pi^2(4) = \min(7+4, +\infty) = 11$
$\pi^2(5) = \min(7+1, +\infty) = 8$
$\pi^2(6) = \min(8+1, +\infty) = 9$

k = 3
$\pi^3(2) = 7$

$\pi^2(3) = 8$
$\pi^3(3) = \min(8, 8 + (-2)) = 6$
$\pi^3(4) = \min(11, 8 + 2) = 10$
$\pi^3(5) = 8$
$\pi^3(6) = \min(9, 11-2) = 9$

k = 4
$\pi^4(2) = 7$
$\pi^4(3) = 6$
$\pi^4(4) = 10$
$\pi^4(5) = 8$
$\pi^4(6) = \min(9, 10-2, 6+1) = 7$

k = 5
$\pi^5(2) = 7$
$\pi^5(3) = 6$
$\pi^5(4) = 10$
$\pi^5(5) = 8$
$\pi^5(6) = 7$

3.6 **Quelques algo pour les graphes sans circuit**.

- Si G est sans circuit et si l'ordre des sommets est bien choisi, alors l'algorithme de Bellman converge en une seule fois.

- Supposons que le sommet 1 soit le seul sommet tel que $\Gamma^{-1}=\varnothing$. Ce sommet est appelé la **racine** du graphe.

Algorithme

a) Poser $\pi(1) = 0$ et $CC = \{1\}$

b) TANT QUE $|CC| \neq |S|$ FAIRE

Chercher un sommet $j \in S \backslash CC$ tel que $\Gamma^{-1}(j) \subset CC$

Poser $\pi(j) = \min_{i \in \Gamma^{-1}(j)}(\pi(i) + l_{ij})$

$CC \leftarrow CC \cup \{j\}$

Ex. (celui de Bellman)

INIT : CC = {1} π(1) = 0

J=2
π(2) = 7
CC = {1, 2}

J = 5
π(5) = 8
CC = {1, 2, 5}

J = 3
π(3) = min(8, 8 –2) = 6
CC = {1, 2, 5, 3}

J = 4
π(4) = min(7+4 , 8+2)=10
CC = {1, 2, 5, 3, 4}

J = 6
π(6) = min(6+1,10-2) = 7
CC = {1, 2, 5, 3, 4, 6}

Recherche de la fonction rang

- Cette fonction est très utile pour classer les sommets suivant un ordre bien précis
- L'intérêt de ce classement : **convergence** des algorithmes.
- Elle est définie comme suit :

  r(1) = 0

  r(i) = nombre d'arcs dans un chemin de cardinalité **maximum** entre 1 et i.

Evidemment, cette fonction n'est pas applicable à un graphe ayant un circuit.

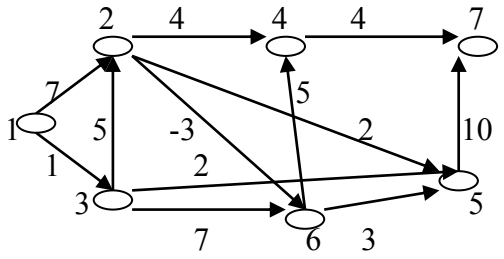
Algorithme

a) $\forall i \in S\ d^{-1}(i) = |\Gamma^{-1}(i)|$; k = 0 ; $S_0 = \{1\}$ /*l'ensemble des sommets dont le rang est 0*/

b) $S_{k+1} \leftarrow \varnothing$

POUR TOUT $i \in S_k$ FAIRE
    r(i) ← k
    POUR TOUT $j \in \Gamma(i)$ FAIRE
        $d^{-1}(j) \leftarrow d^{-1}(j) - 1$
        SI $d^{-1}(j) = 0$ ALORS $S_{k+1} \leftarrow S_{k+1} \cup \{j\}$
    k ← k+1

Ex.



r(1) = 0
j=2
$d^{-1}(2) = 1$
j=3
$d^{-1}(3) = 0$
$S_1 = \{3\}$

r(3) = 1
j=2
$d^{-1}(2) = 0$
$S_2 = \{2\}$
j=5
$d^{-1}(5) = 2$
j=6
$d^{-1}(6) = 1$
k=2

r(2) = 2
j=4
$d^{-1}(4) = 1$
j=5
$d^{-1}(5) = 1$
j=6
$d^{-1}(6) = 0$
$S_3 = \{6\}$
k=3

r(6) = 3
j=4
$d^{-1}(4) = 0$
$S_4 = \{4\}$
j=5
$d^{-1}(5) = 0$
$S_4 = \{4, 5\}$
k←4

r(4) = 4
j=7
$d^{-1}(7) = 1$
r(5) = 4
j=7
$d^{-1}(7) = 0$

r(7) = {5}

3.7 **Algorithmes matriciels : Algorithme de Floyd**

Ils interviennent dans le cas de la recherche d'un plus court chemin entre tous les couples de sommets.

On considère successivement les chemins de i vers j qui ne passent d'abord par aucun autre sommet, puis qui passent éventuellement par le sommet 1, puis par les sommets 1 et 2 etc. A l'étape k, on calcule le coût $d_k(i,j)$ d'un plus court chemin de i à j passant par des sommets inférieurs ou égaux à k. On note $pred_k(i,j)$ le prédécesseur de j dans ce plus court chemin.

On utilise une matrice carrée L d'ordre N, initialiseé aux valeurs

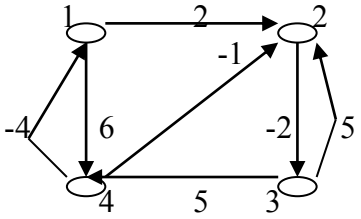
l_{ij} = longueur de l'arc (i,j) si (i,j) $\in$ A
$l_{ii} = 0$
$+\infty$ sinon

On utilise également une matrice carrée P d'ordre N telle que P[i,j]=pred(i,j), s'il existe un chemin de i vers j.

Algorithme
FLOYD(G ; L, P)
E : G : Graphe
S : L = tableau[1..n, 1..n] de réel, P = tableau[1..n, 1..n] d'ent.
{Le graphe G est valué par des coûts quelconques ; on suppose qu'il n'y a pas de circuit absorbant ; on cherche un plus court chemin entre deux sommets quelconques}

```
$1
{init}
POUR i ← 1 à n FAIRE
        $11
        POUR j ← 1 à n FAIRE
                $111
                L[i,j] ← coût(i,j,G)
                P[i,j] ←i
                111$
        11$
{calcul des plus courts chemins}
POUR k ← 1 à N FAIRE
        $12
        POUR i ← 1 à N FAIRE
                $121
                POUR j ← 1 à N FAIRE
                        $1211
                        SI L[i,k] + L[k,j] < L[i,j] ALORS
                                $12111
                                L[i,j] ← L[i,k] + L[k,j]
                                P[i,j] ← P[k,j]
                                12111$
                        1211$
                121$
        12$
1$
```

Exemple



INIT.

	1	**2**	**3**	**4**			**1**	**2**	**3**	**4**	
	0	2	∞	6	|**1**		1	1	1	1	|**1**
L=	∞	0	-2	∞	|**2**	P=	2	2	2	2	|**2**
	∞	5	0	5	|**3**		3	3	3	3	|**3**
	-4	-1	∞	0	|**4**		4	4	4	4	|**4**

k=1

	0	2	∞	6		1	1	1	1
L=	∞	0	-2	∞	P=	2	2	2	2
	∞	5	0	5		3	3	3	3
	-4	-2	∞	0		4	1	4	4

k=2

$$L=\begin{pmatrix} 0 & 2 & 0 & 6 \\ \infty & 0 & -2 & \infty \\ \infty & 5 & 0 & 5 \\ -4 & -2 & -4 & 0 \end{pmatrix} \qquad P=\begin{pmatrix} 1 & 1 & 2 & 1 \\ 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 \\ 4 & 1 & 2 & 4 \end{pmatrix}$$

k=3

$$L=\begin{pmatrix} 0 & 2 & 0 & 5 \\ \infty & 0 & -2 & 3 \\ \infty & 5 & 0 & 5 \\ -4 & -2 & -4 & 0 \end{pmatrix} \qquad P=\begin{pmatrix} 1 & 1 & 2 & 3 \\ 2 & 2 & 2 & 3 \\ 3 & 3 & 3 & 3 \\ 4 & 1 & 2 & 4 \end{pmatrix}$$

k=4

$$L=\begin{pmatrix} 0 & 2 & 0 & 5 \\ -1 & 0 & -2 & 3 \\ 1 & 4 & 0 & 5 \\ -4 & -2 & -4 & 0 \end{pmatrix} \qquad P=\begin{pmatrix} 1 & 1 & 2 & 3 \\ 4 & 2 & 2 & 3 \\ 4 & 4 & 3 & 3 \\ 4 & 1 & 2 & 4 \end{pmatrix}$$