

Chemins de poids minimum

F.M.

2024/2025

Graphe pondéré

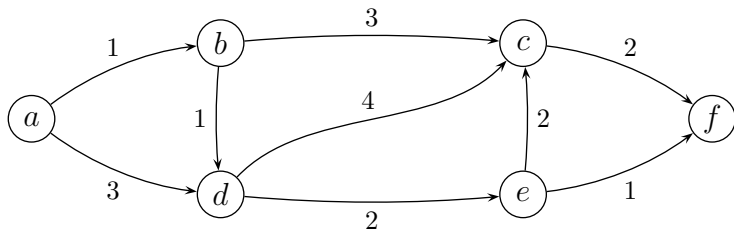
Un **graphe pondéré** est un triplet (S, A, p) où :

- S est un ensemble de sommets,
- A est un ensemble d'arcs (ou d'arêtes),
- p est une application appelée **pondération** qui à tout arc (ou arête) $u \in A$ associe le nombre $p(u) \in \mathbb{R}$ appelé **poids** de u .

Les poids peuvent représenter des distances, des durées, des coûts, des gains ou des pertes, des capacités, des probabilités, etc.

Dans ce chapitre, le terme graphe désigne un graphe orienté, simple, connexe et pondéré.

Diagramme d'un graphe pondéré



$$p(a, b) = 1$$

$$p(a, d) = 3$$

$$p(b, c) = 3$$

$$p(b, d) = 1$$

$$p(c, f) = 2$$

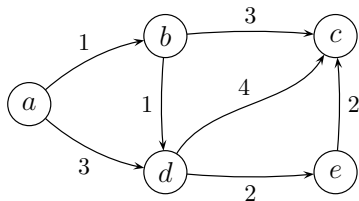
$$p(d, c) = 4$$

$$p(d, e) = 2$$

$$p(e, c) = 2$$

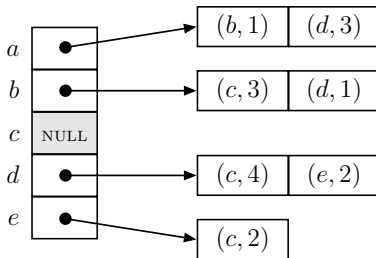
$$p(e, f) = 1$$

Représentation des graphes pondérés



	a	b	c	d	e
a	0	1	∞	3	∞
b	∞	0	3	1	∞
c	∞	∞	0	∞	∞
d	∞	∞	4	0	2
e	∞	∞	2	∞	0

matrice des poids



liste d'adjacence

Poids d'un chemin

Le **poids d'un chemin** dans un graphe $G = (S, A, p)$ est égal à **la somme des poids des arcs** qui le composent. On note $P(C)$ le poids d'un chemin C .

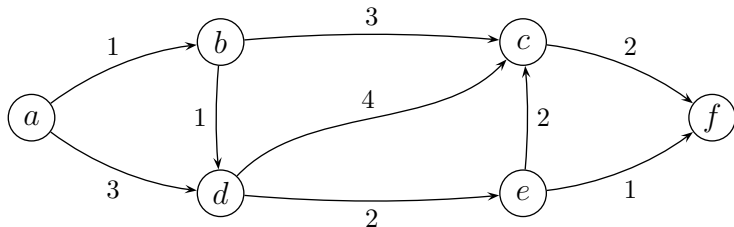
Le poids d'un chemin (s_1, s_2, \dots, s_k) est donc :

$$P(s_1, s_2, \dots, s_k) = p(s_1, s_2) + p(s_2, s_3) + \dots + p(s_{k-1}, s_k)$$

Cas particuliers :

- Si $p(u) = 1$ pour tout arc $u \in A$ alors le poids d'un chemin est égal à sa longueur.
- On convient que le poids d'un chemin de longueur 0 (c.-à-d. un chemin réduit à un seul sommet) est nul.

Illustration

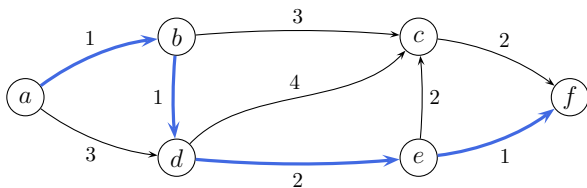


$$P(a, b, d) = p(a, b) + p(b, d) = 2 \quad \ell(a, b, d) = 2$$

$$P(a, b, d, c) = p(a, b) + p(b, d) + p(d, c) = 6 \quad \ell(a, b, d, c) = 3$$

Chemin de poids minimum

Un chemin C allant d'un sommet x à un sommet y dans un graphe G est **un chemin de poids minimum** (cpm en abrégé) ou **un plus court chemin** (pcc en abrégé) si et seulement si tout autre chemin allant de x à y dans G a un poids supérieur ou égal à celui de C . Il peut exister plusieurs cpm allant de x à y (il n'a pas unicité).



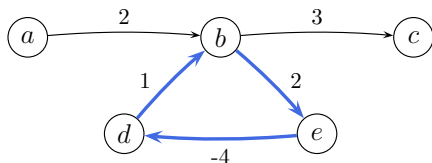
$$\begin{aligned} P(a, b, c, f) &= 6 & P(a, b, d, c, f) &= 8 & P(a, b, d, e, c, f) &= 8 \\ P(a, b, d, e, f) &= 5 & P(a, d, c, f) &= 9 & P(a, d, e, c, f) &= 9 & P(a, d, e, f) &= 6 \end{aligned}$$

(a, b, d, e, f) est un cpm (ou un pcc) allant de a à f

Problème : déterminer un cpm allant d'un sommet x à un sommet y

Circuit absorbant et conséquences

Un circuit de poids négatif est dit **absorbant**.



(b, e, d, b) est un circuit absorbant de poids -1

Quel est le chemin de poids minimum allant de a à c ?

$$P(a, b, c) = 5$$

$$P(a, b, e, d, b, c) = 5 - 1 = 4 \quad P(a, b[e, d, b]^{\times k}, c) = 5 - k$$

Dans cette situation, on dira qu'il n'existe aucun chemin de poids minimum allant de a vers c .

Si un graphe contient des circuits absorbants, il est possible qu'il n'existe aucun chemin de poids minimum entre certains sommets.

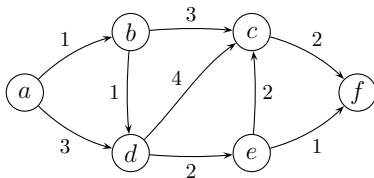
Si un graphe ne contient pas de circuit absorbant alors tout chemin de poids minimum est nécessairement élémentaire.

Distance

La **distance** d'un sommet x à un sommet y est le poids d'un plus court chemin allant de x à y . On la note **$\text{DIST}(x, y)$** .

Cas particuliers :

- Pour tout sommet x , **$\text{DIST}(x, x) = 0$** .
- Si il n'existe aucun chemin de x à y , on pose **$\text{DIST}(x, y) = +\infty$** .
- Si il existe un chemin de x à y passant par un circuit absorbant alors **$\text{DIST}(x, y) = -\infty$** .



$$\text{DIST}(a, c) = 4$$

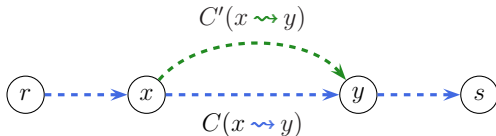
$$\text{DIST}(a, f) = 5$$

$$\text{DIST}(d, b) = +\infty$$

Propriété d'optimalité

Soit $G = (S, A, p)$ un graphe **sans circuit absorbant**.

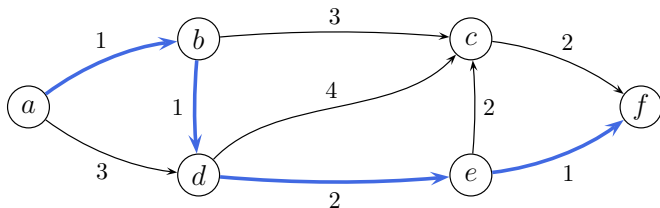
Tout sous-chemin d'un chemin de poids minimum de G est un chemin de poids minimum de G .



Soient $C(r \rightsquigarrow s)$ un cpm allant du sommet r au sommet s dans G et $C(x \rightsquigarrow y)$ le sous-chemin de ce chemin allant du sommet x au sommet y .

Supposons qu'il existe un chemin $C'(x \rightsquigarrow y)$ de poids inférieur à celui de $C(x \rightsquigarrow y)$. Alors, en remplaçant $C'(x \rightsquigarrow y)$ par $C(x \rightsquigarrow y)$ dans $C(r \rightsquigarrow s)$, on obtient un chemin de r à s dont le poids est inférieur à celui de $C(r \rightsquigarrow s)$: contradiction, puisque $C(r \rightsquigarrow s)$ est un cpm.

Illustration



(a, b, d, e, f) est un cpm de a à f

(a, b) , (a, b, d) , (a, b, d, e) , (b, d) , (b, d, e) , (b, d, e, f) ,
 (d, e) , (d, e, f) et (e, f) sont aussi des cpm

Un algorithme qui détermine un chemin de poids minimum allant d'un sommet x à un sommet y détermine en même temps plusieurs autres chemins de poids minimum.

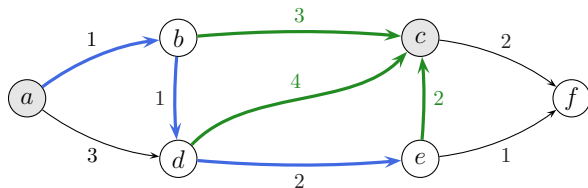
Soit $G = (S, A, p)$ un graphe **sans circuit absorbant** et soient r et x deux sommets de G .

Un pcc de r à x contient un pcc de r vers un prédécesseur de x . On peut donc calculer la distance entre r et x quand les distances entre r et les prédécesseurs de x sont connues :

$$\text{DIST}(r, x) = \min\{\text{DIST}(r, s) + p(s, x), s \in V^-(x)\}$$

```
DIST( $r, x$ )  $\leftarrow +\infty$ 
pour tout  $s \in V^-(x)$  faire
    |   si  $\text{DIST}(r, s) + p(s, x) < \text{DIST}(r, x)$  alors
    |       |    $\text{DIST}(r, x) \leftarrow \text{DIST}(r, s) + p(s, x)$ 
    |   fin si
fin pour
```

Illustration



$$V^-(c) = \{b, d, e\}$$

Il s'agit de calculer $\text{DIST}(a, c)$ sachant que $\text{DIST}(a, b) = 1$, $\text{DIST}(a, d) = 2$ et $\text{DIST}(a, e) = 4$.

$$\text{DIST}(a, b) + p(b, c) = 1 + 3 = 4$$

$$\text{DIST}(a, d) + p(d, c) = 2 + 4 = 6$$

$$\text{DIST}(a, e) + p(e, c) = 4 + 2 = 6$$

$\text{DIST}(a, c) = 4$ et (a, b, c) est un cpm de a à c

Soit G un graphe pondéré **sans circuit absorbant** et soit r un sommet donné de G .

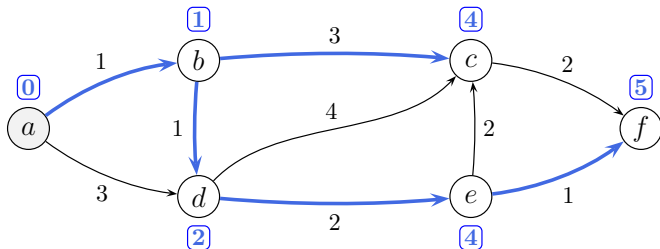
Il existe une r -arborescence contenant tous les descendants de r et telle que tout chemin de cette arborescence est de poids minimum. De plus, tout arc (s, x) du graphe vérifie alors :

$$\text{DIST}(r, s) + p(r, x) \geq \text{DIST}(r, x)$$

Cette r -arborescence est appelée r -arborescence de chemins de poids minimum.

Remarques : pour chaque descendant s de r , cette r -arborescence contient un chemin de poids minimum allant de r à s ; il peut exister plusieurs r -arborescences de chemins de poids minimum (il n'y a pas unicité).

Illustration



(a, b) est un cpm de a à b

(a, b, c) est un cpm de a à c

(a, b, d) est un cpm de a à d

(a, b, d, e) est un cpm de a à e

(a, b, d, e, f) est un cpm de a à f

a -arborescence de chemins de poids minimum $\rightarrow \boxed{\text{DIST}(a, s)}$

tout arc (s, x) vérifie : $\text{DIST}(a, s) + p(s, x) \geq \text{DIST}(a, x)$

Le problème

Problème du pcc : trouver un chemin de poids minimum allant d'un sommet x vers un sommet y dans un graphe pondéré.

Un algorithme non envisageable consiste à énumérer tous les chemins de x à y et à calculer leur poids. Par exemple, un graphe complet d'ordre 20 contient environ 17×10^{15} chemins élémentaires entre 2 sommets donnés.

Il est plus efficace de résoudre le problème suivant :

Problème des pcc à origine unique : étant donné un sommet r , déterminer une r -arborescence de chemins de poids minimum. C'est ce problème qui sera traité dans la suite.

Problème équivalent : étant donné un sommet r , calculer $\text{DIST}(r, s)$ pour chaque sommet s du graphe.

Structure de données pour les arborescences

Soit r un sommet d'un graphe G et soit T une r -arborescence (T est un sous-graphe partiel de G). Pour représenter T , on utilise deux tables.

- Une **table des pères** \mathcal{A} pour mémoriser les arcs appartenant à l'arborescence : $\forall s \in S$,

$$\mathcal{A}[s] = \begin{cases} \emptyset & \text{si } s = r \\ x & \text{si l'arc } (x, s) \in T \\ s & \text{si } s \notin T \end{cases}$$

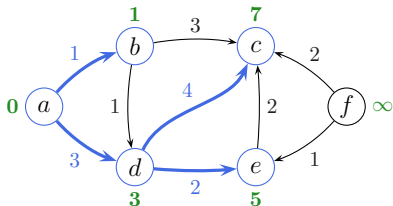
- Une **table des potentiels** Π pour mémoriser les poids des chemins d'origine r dans l'arborescence : $\forall s \in S$,

$$\Pi[s] = \begin{cases} \text{poids du chemin de } r \text{ à } s \text{ dans } T & \text{si } s \in T \\ \infty & \text{si } s \notin T \end{cases}$$

Le nombre $\Pi[s]$ est appelé **potentiel** du sommet s .

Illustration

Voici une a -arborescence :



Tout arc (x, y) de l'arborescence vérifie : $\Pi[x] + p(x, y) = \Pi[y]$

Et sa représentation :

s	a	b	c	d	e	f
$\mathcal{A}[s]$	\emptyset	a	d	a	d	f
$\Pi[s]$	0	1	7	3	5	∞

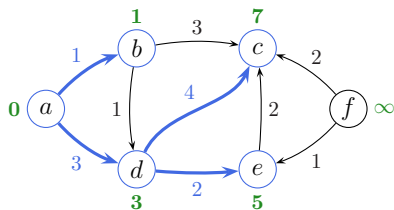
Ce n'est pas une a -arborescence de cpm. Par exemple, le chemin de l'arborescence allant de a à c est (a, d, c) de poids 7 alors que le cpm allant de a à c est (a, b, c) de poids 4.

Arc améliorant

Soit $G = (S, A)$ un graphe et T une r -arborescence de G représentée par les tables \mathcal{A} et Π .

Un arc $(s, x) \in A$ est dit améliorant si $\Pi[s] + p(s, x) < \Pi[x]$.

Un arc (s, x) est améliorant si le chemin de T allant de r à s suivi par l'arc (s, x) est plus court que le chemin de T allant de r à x . Autrement dit, en passant par l'arc (s, x) on obtient un meilleur chemin allant de r vers x .



(b, d) et (b, c) sont améliorants :

$$\Pi[b] + p(b, d) = 2 < \Pi[d] = 3$$

$$\Pi[b] + p(b, c) = 4 < \Pi[c] = 7$$

Si G contient des arcs améliorants alors T n'est pas une r -arborescence de chemins de poids minimum.

Propriété fondamentale

Soit $G = (S, A)$ un graphe et T une r -arborescence de G représentée par les tables \mathcal{A} et Π .

Alors T est une r -arborescence de chemins de poids minimum si et seulement si G ne contient aucun arc améliorant.

Autrement dit, T est une r -arborescence de chemins de poids minimum si et seulement si :

$$\forall (s, x) \in A, \Pi[s] + p(s, x) \geq \Pi[x]$$

Puisque les potentiels sont égaux aux distances quand T est une r -arborescence de cpm.

Preuve de la propriété fondamentale

(\Rightarrow) Si T est une r -arborescence de cpm alors pour tout sommet x de T , $\Pi[x] = \text{DIST}(r, x)$. Les inégalités sont alors vérifiées (d'après la propriété d'optimalité).

(\Leftarrow) Soit T une r -arborescence vérifiant les inégalités. Considérons un chemin $\mathcal{C} = (r, s_1, \dots, s_k, x)$ allant de r à x dans G . Alors les arcs de \mathcal{C} vérifient :

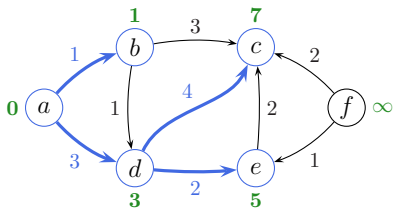
$$\begin{aligned}\Pi[r] + p(r, s_1) &\geq \Pi[s_1] \\ \Pi[s_1] + p(s_1, s_2) &\geq \Pi[s_2] \\ &\dots \geq \dots \\ \Pi[s_k] + p(s_k, x) &\geq \Pi[x]\end{aligned}$$

La somme de ces inégalités donne :

$$\begin{aligned}\Pi[r] + p(r, s_1) + p(s_1, s_2) + \dots + p(s_k, x) &\geq \Pi[x] \\ \Pi[r] + P(\mathcal{C}) &\geq \Pi[x]\end{aligned}$$

Puisque $\Pi[r] = 0$ on obtient : $P(\mathcal{C}) \geq \Pi[x]$. Le chemin de r à x dans T est donc un cpm de r à x dans G .

Illustration de la propriété fondamentale

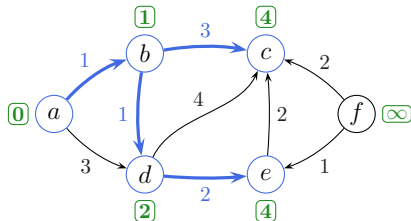


(b, c) et (b, d) sont améliorants :

$$\Pi[b] + p(b, d) = 2 < \Pi[d] = 3$$

$$\Pi[b] + p(b, c) = 4 < \Pi[c] = 7$$

Cette a -arborescence n'est donc pas une a -arborescence de cpm.



Tout arc (x, y) du graphe est tel que $\Pi[x] + p(x, y) \geq \Pi[y]$: il n'y a aucun arc améliorant.




Cette arborescence est donc une a -arborescence de cpm et les potentiels sont des distances.

Les algorithmes reposent sur la propriété fondamentale : ils construisent progressivement une arborescence qui vérifie cette propriété.

Procédure fondamentale : le relâchement d'un arc

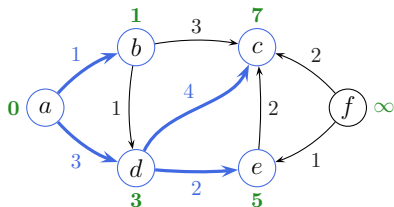
Soit $G = (S, A, p)$ un graphe pondéré et soit T une r -arborescence représentée par les tables Π et \mathcal{A} .

La **procédure de relâchement** ou de **relaxation d'un arc (s, x)** consiste à **tester si l'arc (s, x) est améliorant** et le cas échéant, à **modifier l'arborescence pour y intégrer l'arc (s, x)** .

```
proc relâcher( $s, x$ )  
  si  $\Pi[s] + p(s, x) < \Pi[x]$  alors .....  si  $(s, x)$  est améliorant  
    |  
    |  $\Pi[x] \leftarrow \Pi[s] + p(s, x)$  .....  amélioration de  $\Pi[x]$   
    |  $\mathcal{A}[x] \leftarrow s$  .....   $s$  devient le père de  $x$   
  fin si  
fin proc
```

Si (s, x) est améliorant, cette procédure remplace l'arc de but x se trouvant dans T par l'arc (s, x) . Elle n'a aucun effet sinon.

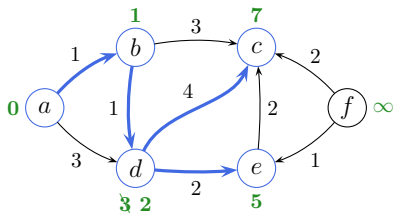
Illustration



s	a	b	c	d	e	f
$\Pi[s]$	0	1	7	3	5	∞
$\mathcal{A}[s]$	\emptyset	a	d	a	d	f

(b, d) et (b, c) sont améliorants

relâcher(b, d) a pour effet de remplacer l'arc (a, d) de l'arborescence par l'arc (b, d) :



s	a	b	c	d	e	f
$\Pi[s]$	0	1	7	2	5	∞
$\mathcal{A}[s]$	\emptyset	a	d	b	d	f

(b, c) est toujours améliorants

$\Pi[d]$ est amélioré et les arcs (d, c) et (d, e) sont devenus améliorants

Comment poursuivre ?

Algorithme de base : initialisations

Soit $G = (S, A, p)$ un graphe pondéré **sans circuit absorbant** et soit r un sommet de G . L'algorithme de base détermine **une r -arborescence de chemins de poids minimum**.

Le point de départ de l'algorithme est une arborescence triviale réduite au seul sommet r dont le potentiel est fixé à 0.

```
pour tout  $s \in S$  faire  
   $\Pi[s] \leftarrow \infty; \mathcal{A}[s] \leftarrow s$   
fin pour  
 $\Pi[r] \leftarrow 0; \mathcal{A}[r] \leftarrow \emptyset$ 
```

Le potentiel de tout sommet $s \neq r$ est fixé à ∞ (une grande valeur).

En l'absence de circuits absorbants, la valeur $\Pi[r] = 0$ est optimale et ne sera plus modifiée par la suite. On a donc : $\text{DIST}(r, r) = \Pi[r] = 0$.

Algorithme de base : les itérations

L'algorithme construit progressivement l'arborescence de la façon suivante :

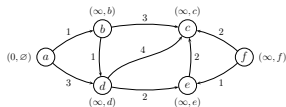
```
tant que  $\exists (s, x) \in A$  tel que  $\Pi[s] + p(s, x) < \Pi[x]$  faire  
|    $\Pi[x] \leftarrow \Pi[s] + p(s, x)$  ;  $\mathcal{A}[x] \leftarrow s$   
fin tant que
```

En une phrase : **tant qu'il existe un arc améliorant, intégrer cet arc à l'arborescence.**

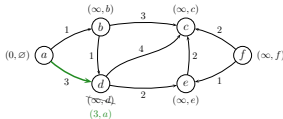
L'arrêt se produit quand il n'existe plus d'arc améliorant, c.-à-d. quand $\Pi[s] + p(s, x) \geq \Pi[x]$ pour tout arc (s, x) du graphe. La **propriété fondamentale** (cf. diapo 21) permet alors de conclure que l'arborescence obtenue est bien une r -arborescence de cpm et que $\Pi[s] = \text{DIST}(r, s)$ pour tout sommet s du graphe.

La terminaison de l'algorithme est garantie en l'absence de circuits absorbants : **les potentiels convergent vers les distances.**

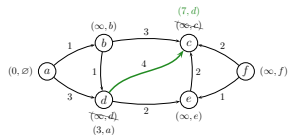
Illustration de l'algorithme de base



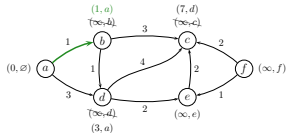
Initialisations quand $r = a$



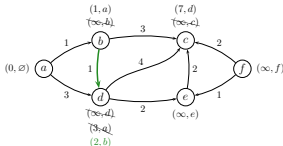
(a, d) est améliorant
 $\Pi[d] \leftarrow 3$ et $\mathcal{A}[d] \leftarrow a$



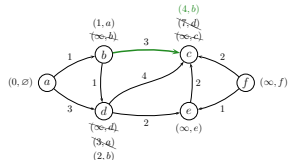
(d, c) est améliorant
 $\Pi[c] \leftarrow 7$ et $\mathcal{A}[c] \leftarrow d$



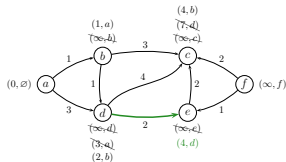
(a, b) est améliorant
 $\Pi[b] \leftarrow 1$ et $\mathcal{A}[b] \leftarrow a$



(b, d) est améliorant
 $\Pi[d] \leftarrow 2$ et $\mathcal{A}[d] \leftarrow b$

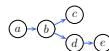
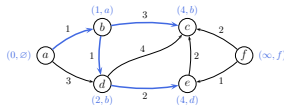


(b, c) est améliorant
 $\Pi[c] \leftarrow 4$ et $\mathcal{A}[c] \leftarrow b$



(d, e) est améliorant
 $\Pi[e] \leftarrow 4$ et $\mathcal{A}[e] \leftarrow d$

il n'y a plus d'arc améliorant et le résultat est :



s	a	b	c	d	e	f
$\Pi[s] = \text{DIST}(a, s)$	0	1	4	2	4	∞
$\mathcal{A}[s]$	\emptyset	a	b	b	d	f

Remarques

L'algorithme de base présenté ici est un algorithme que l'on exécute à *la main* sur le diagramme d'un graphe de petite taille. À chaque itération, un arc est choisi arbitrairement parmi les arcs améliorants et il est relâché. Le résultat final ne dépend pas du choix effectué à chaque itération.

En l'état, l'algorithme de base ne fonctionne pas si le graphe contient des circuits absorbants car il sera toujours possible de trouver un arc améliorant dans ce cas (l'algorithme ne se termine pas). Il faudrait ajouter un test qui détecte les circuits absorbants.

Dans la suite, on présente les algorithmes les plus connus et nous verrons comment les implémenter. Ils reposent tous sur la même idée : relâcher des arcs jusqu'à ce qu'il n'y ait plus d'arcs améliorants.