Algorithme de Bellman

F.M.

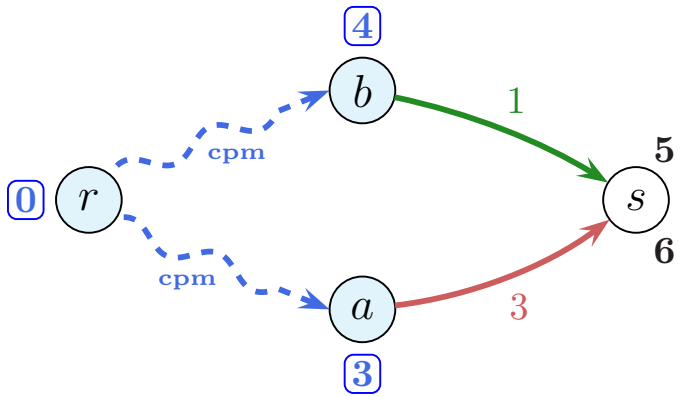
2024/2025

Rappel : propriété d'optimalité

Soient $G = (S, A, p)$ un graphe pondéré sans circuit absorbant et r un sommet de G.

$$\textsc{Dist}(r, s) = \min \left\{ \textsc{Dist}(r, x) + p(x, s) \ ; \ x \in V^-(s) \right\}$$

On peut calculer $\textsc{Dist}(r, s)$ à partir des $\textsc{Dist}(r, x)$, $x \in V^-(s)$.



on connaît :

$\textsc{Dist}(r, a) = 3$

$\textsc{Dist}(r, b) = 4$

on calcule :

$$\begin{array}{|llllllll} \textsc{Dist}(r, a) & + & p(a, s) & = & 3 & + & 3 & = & 6 \\ \textsc{Dist}(r, b) & + & p(b, s) & = & 4 & + & 1 & = & 5 \end{array}$$

Propriété d'optimalité $\Rightarrow$ $\textsc{Dist}(r, s) = 5$

2 / 25

Propriété d'optimalité et ordre topologique

Soit L un ordre topologique pour G. Rappelons que tous les prédécesseurs d'un sommet s se trouvent à gauche de s dans L). On peut alors utiliser la propriété d'optimalité pour déterminer une r-arborescence de cpm :

✎ Initialisations

$$\forall s \in S,\ \Pi[s] \leftarrow \begin{cases} 0 & \text{si } s = r \\ \infty & \text{si } s \neq r \end{cases} \quad \text{et} \quad \mathcal{A}[s] \leftarrow \begin{cases} \varnothing & \text{si } s = r \\ s & \text{si } s \neq r \end{cases}$$

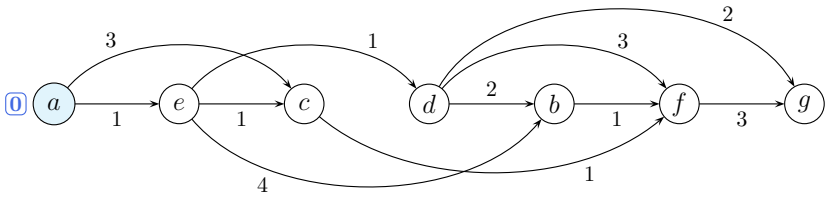
✎ Itérations : une seule phase de relâchements

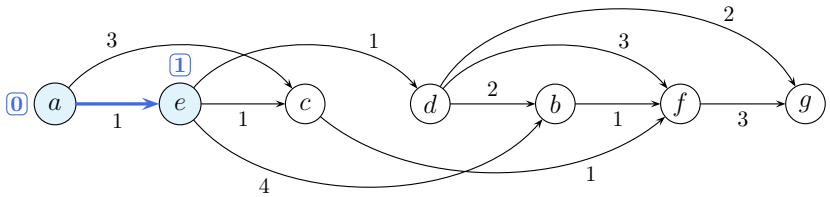
pour tout $s \in L$ **faire**
  ✎ relâcher tous les arcs de but $s \Leftrightarrow$ appliquer la propriété
  pour tout $x \in V^-(s)$ **faire**
    relacher(x,s)
  fin pour
  ✎ propriété $\Rightarrow \Pi[s] = \textsc{Dist}(r, s)$
fin pour

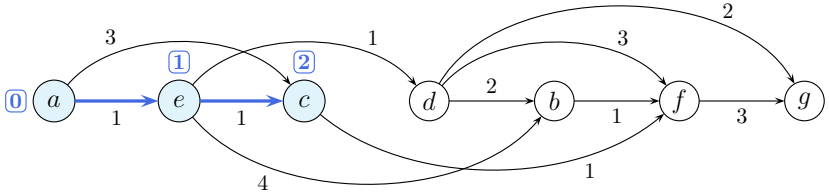
C'est une première version de l'algorithme de Bellman : pour chaque sommet s pris dans un ordre topologique, on relâche tous les arcs de but s.

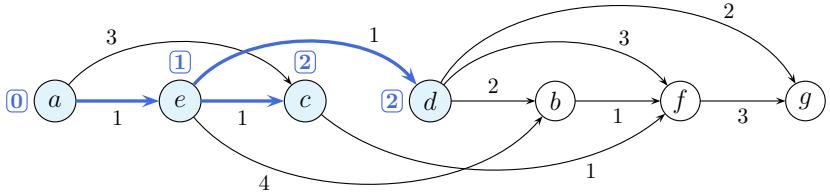
3 / 25

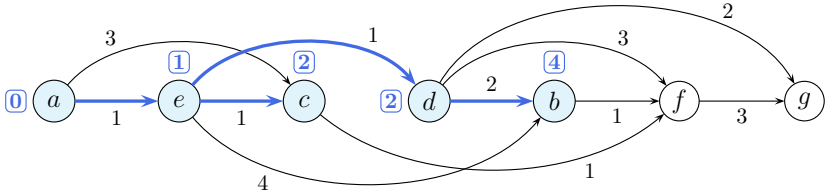
Illustration

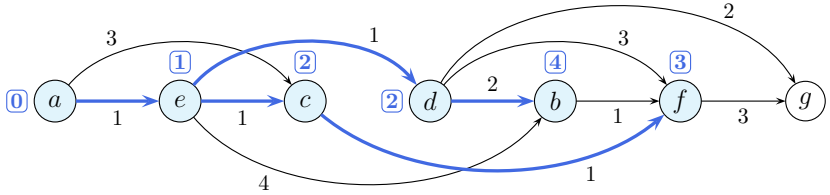


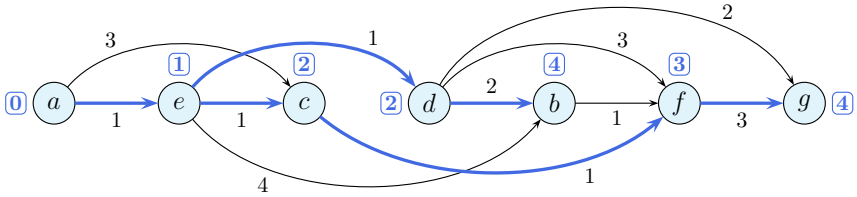












Première version de l'algorithme de Bellman qui repose sur la propriété d'optimalité.

4 / 25

Remarques

Cette première version de l'algorithme de Bellman nécessite de connaitre **l'ensemble des prédécesseurs de chaque sommet du graphe**.

La représentation généralement utilisée est la **représentation par liste d'adjacence** qui associe à chaque sommet la liste de ses **successeurs**. Pour appliquer cette première version de l'algorithme de Bellman, il est donc nécessaire de déterminer au préalable l'ensemble des prédécesseurs de chaque sommet.

Les algorithmes vus précédemment (Ford, Dijkstra) déterminent des pcc en utilisant la représentation par liste d'adjacence. **Dans la suite, on présente une autre version de l'algorithme de Bellman qui utilise cette structure. C'est cette version qui sera retenue.**

5 / 25

Algorithme de Bellman

Cet algorithme **détermine une r-arborescence de cpm dans un DAG pondéré $G = (S, A, p)$ d'ordre n** représentée par les tables Π (potentiels) et $\mathcal{A}$ (pères).

L'algorithme de Bellman est un algorithme itératif qui examine les sommets **dans un ordre topologique donné $L = (s_1, \ldots, s_k, \ldots, s_n)$. La k^{e} itération consiste à relâcher tous les arcs d'origine s_k**.

Quand arrive l'itération k, tous les arcs de but s_k ont déjà été relâchés (puisque les prédécesseurs de s_k se trouvent à gauche de s_k dans L). La propriété d'optimalité implique alors que $\Pi[s_k] = \textsc{Dist}(r, s_k)$. On relâche ensuite et une fois pour toutes, tous les arcs d'origine s_k.

L'**ordre topologique** permet ici de définir **un ordre de relâchement des arcs** de sorte que, **une et une seule phase de relâchements suffit** pour calculer toutes les distances.

6 / 25

Algorithme de Bellman

✎ Initialisations

$L \leftarrow$ un ordre topologique pour le graphe
pour tout $s \in S$ **faire**
  $\Pi[s] \leftarrow \infty$; $[s] \leftarrow s$
fin pour
$\Pi[r] \leftarrow 0$; $[s] \leftarrow \varnothing$

✎ Itérations : une phase de relâchements

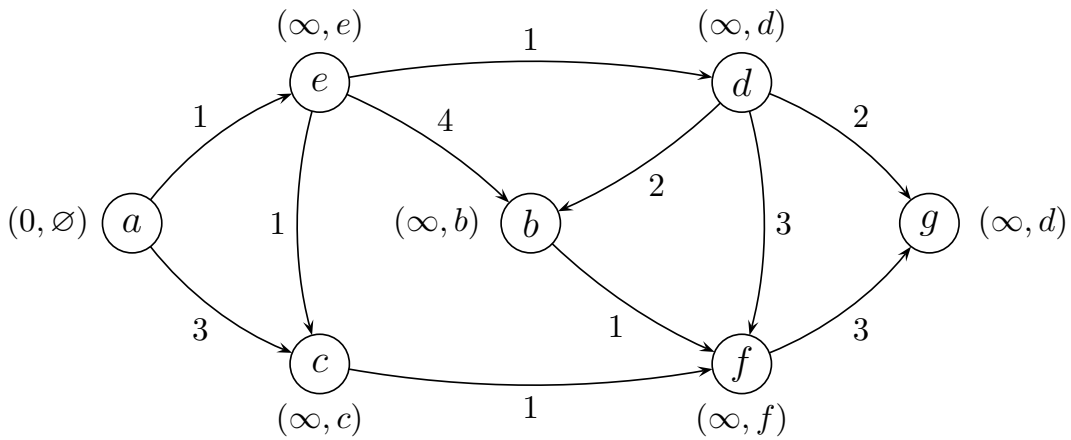
pour tout $s \in L$ **faire**
  ✎ ici $\Pi[s] = \textsc{Dist}(r, s)$ et on relâche tous les arcs d'origine s
  pour tout $x \in V^+(s)$ **faire**
    si $\Pi[s] + p(s, x) < \Pi[x]$ **alors**
      $\Pi[x] \leftarrow \Pi[s] + p(s, x)$; $\mathcal{A}[x] \leftarrow s$
    fin si
  fin pour
fin pour

7 / 25

Trace de l'algorithme de Bellman

Initialisations

$$r = a$$
$$L = (a, e, c, d, b, f, g)$$



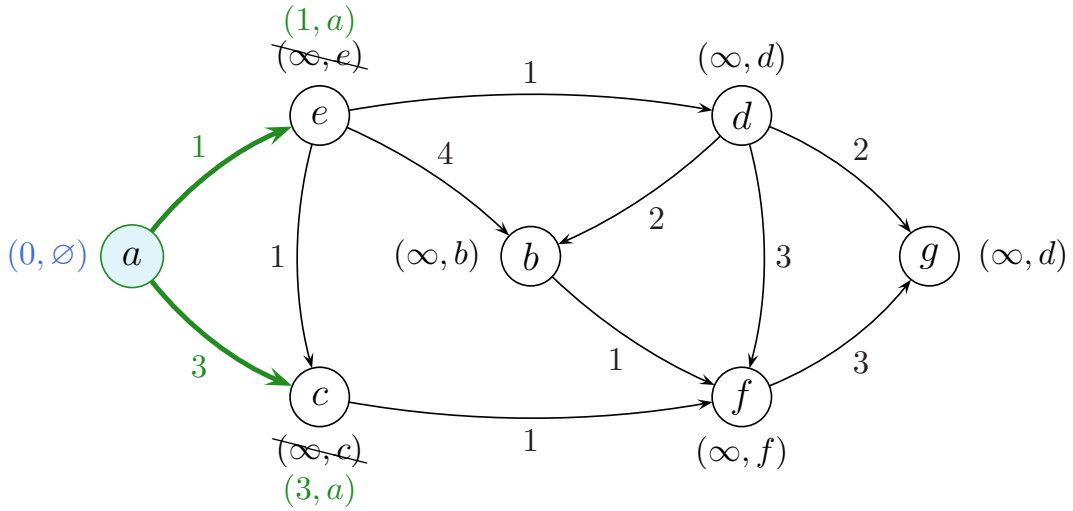
ité.	a	b	c	d	e	f	g
init.	$0, \varnothing$	∞, b	∞, c	∞, d	∞, e	∞, f	∞, g

8 / 25

Trace de l'algorithme de Bellman

Itération 1

$L = (\boldsymbol{a}, e, c, d, b, f, g)$



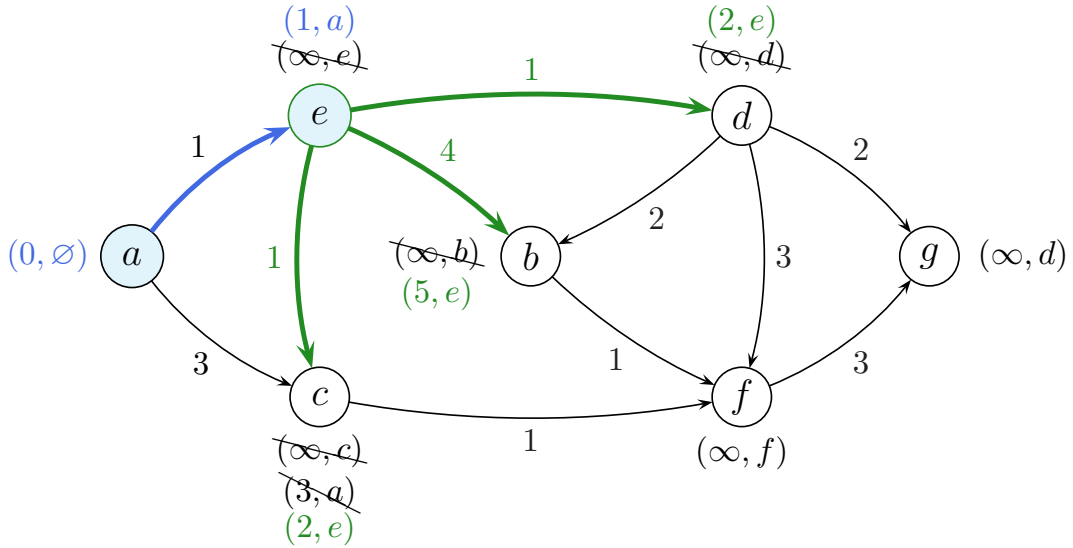
ité.	a	b	c	d	e	f	g
init.	$0, \varnothing$	∞, b	∞, c	∞, d	∞, e	∞, f	∞, g
$\boldsymbol{a}$	$0, \varnothing$	∞, b	$3, a$	∞, d	$1, a$	∞, f	∞, g

9 / 25

Trace de l'algorithme de Bellman

Itération 2

$L = (a, \mathbf{e}, c, d, b, f, g)$



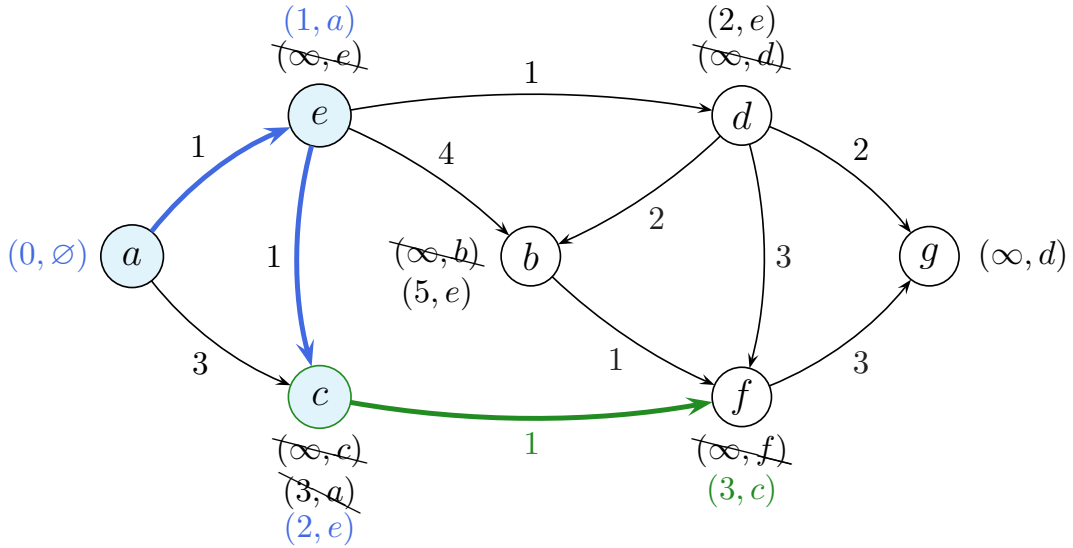
ité.	a	b	c	d	e	f	g
init.	$0, \varnothing$	∞, b	∞, c	∞, d	∞, e	∞, f	∞, g
a	$0, \varnothing$	∞, b	$3, a$	∞, d	$1, a$	∞, f	∞, g
$\mathbf{e}$		$5, e$	$2, e$	$2, e$	$1, a$	∞, f	∞, g

10 / 25

Trace de l'algorithme de Bellman

Itération 3

$L = (a, e, \boldsymbol{c}, d, b, f, g)$



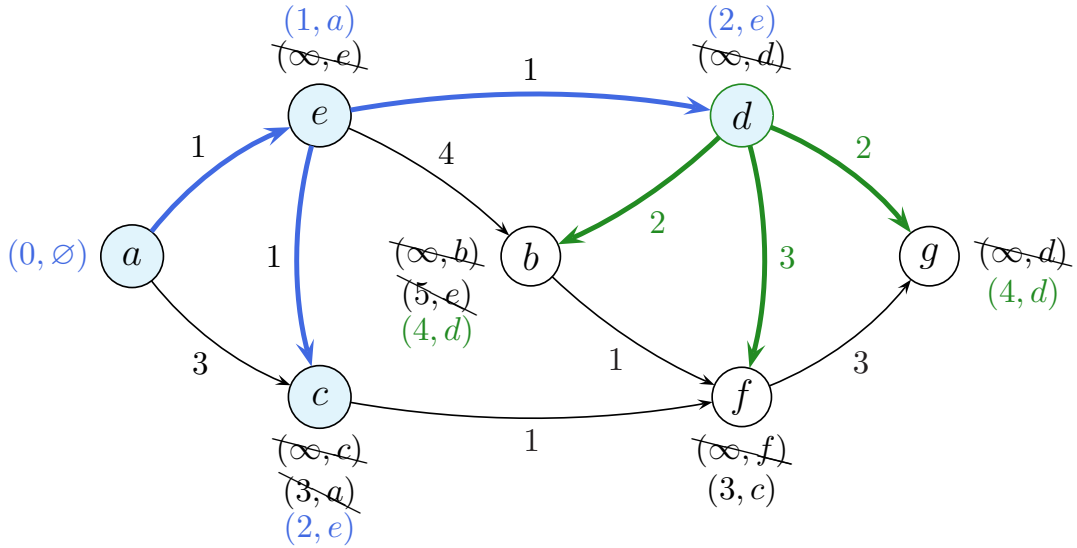
ité.	a	b	c	d	e	f	g
init.	$0, \varnothing$	∞, b	∞, c	∞, d	∞, e	∞, f	∞, g
a	$0, \varnothing$	∞, b	$3, a$	∞, d	$1, a$	∞, f	∞, g
e		$5, e$	$2, e$	$2, e$	$1, a$	∞, f	∞, g
$\boldsymbol{c}$		$5, e$	$2, e$	$2, e$		$3, c$	∞, g

11 / 25

Trace de l'algorithme de Bellman

Itération 4

$L = (a, e, c, \boldsymbol{d}, b, f, g)$



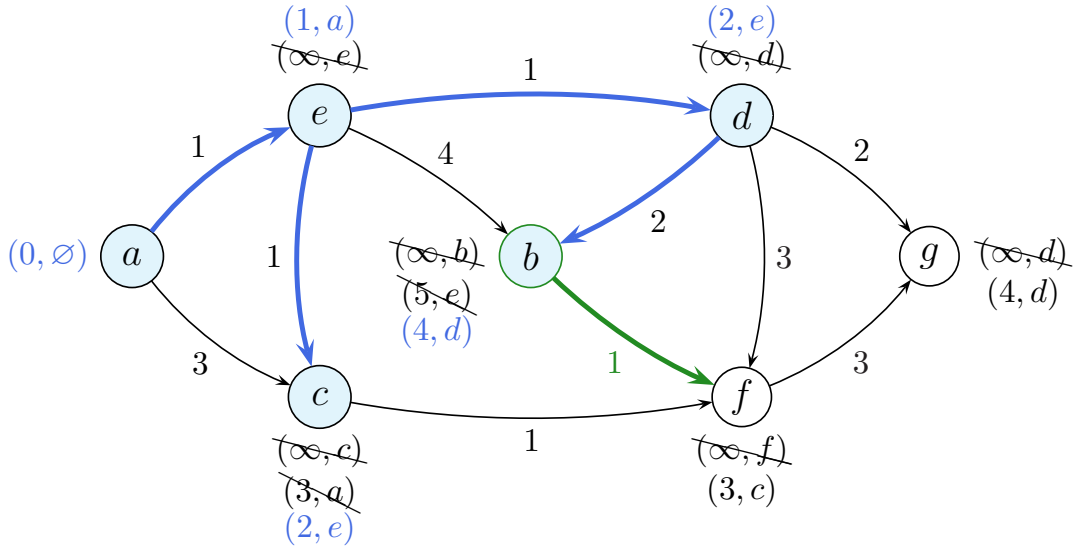
ité.	a	b	c	d	e	f	g
init.	$0, \varnothing$	∞, b	∞, c	∞, d	∞, e	∞, f	∞, g
a	$0, \varnothing$	∞, b	$3, a$	∞, d	$1, a$	∞, f	∞, g
e		$5, e$	$2, e$	$2, e$	$1, a$	∞, f	∞, g
c		$5, e$	$2, e$	$2, e$		$3, c$	∞, g
$\boldsymbol{d}$		$4, d$		$2, e$		$3, c$	$4, d$

12 / 25

Trace de l'algorithme de Bellman

Itération 5

$L = (a, e, c, d, \boldsymbol{b}, f, g)$



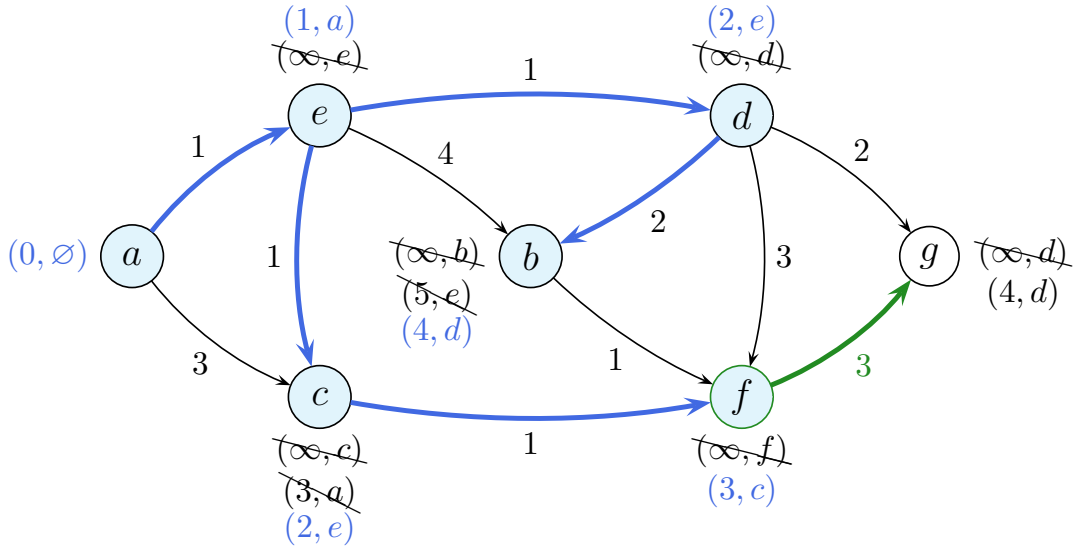
ité.	a	b	c	d	e	f	g
init.	$0, \varnothing$	∞, b	∞, c	∞, d	∞, e	∞, f	∞, g
a	$0, \varnothing$	∞, b	$3, a$	∞, d	$1, a$	∞, f	∞, g
e		$5, e$	$2, e$	$2, e$	$1, a$	∞, f	∞, g
c		$5, e$	$2, e$	$2, e$		$3, c$	∞, g
d		$4, d$		$2, e$		$3, c$	$4, d$
$\boldsymbol{b}$		$4, d$				$3, c$	$4, d$

13 / 25

Trace de l'algorithme de Bellman

Itération 6

$L = (a, e, c, d, b, \boldsymbol{f}, g)$



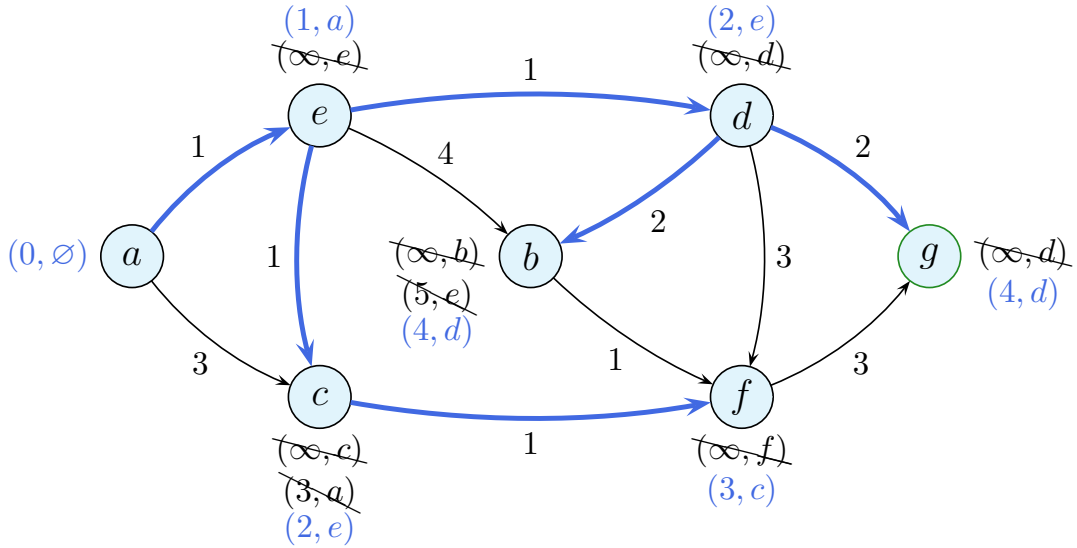
ité.	a	b	c	d	e	f	g
init.	$0, \varnothing$	∞, b	∞, c	∞, d	∞, e	∞, f	∞, g
a	$0, \varnothing$	∞, b	$3, a$	∞, d	$1, a$	∞, f	∞, g
e		$5, e$	$2, e$	$2, e$	$1, a$	∞, f	∞, g
c		$5, e$	$2, e$	$2, e$		$3, c$	∞, g
d		$4, d$		$2, e$		$3, c$	$4, d$
b		$4, d$				$3, c$	$4, d$
$\boldsymbol{f}$						$3, c$	$4, d$

14 / 25

Trace de l'algorithme de Bellman

Itération 7

$L = (a, e, c, d, b, f, \boldsymbol{g})$



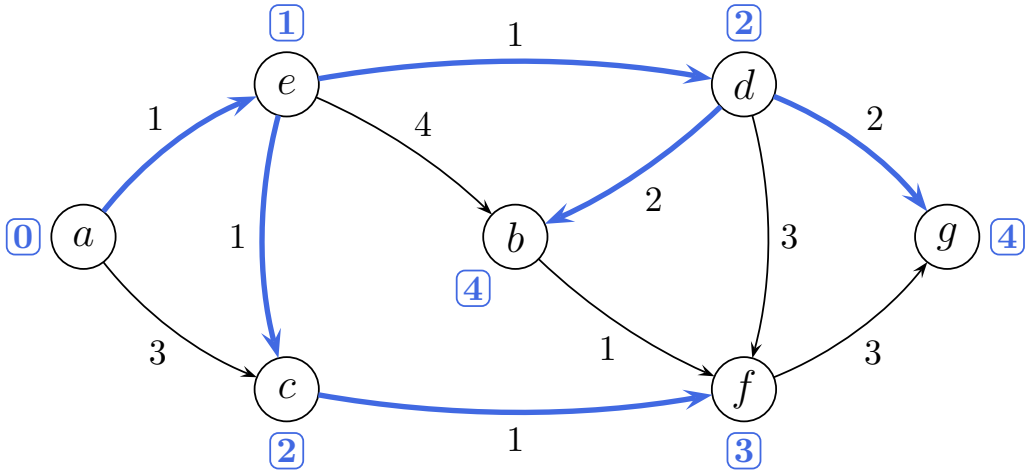
ordre topologique

ité.	a	b	c	d	e	f	g
init.	$0, \varnothing$	∞, b	∞, c	∞, d	∞, e	∞, f	∞, g
a	$0, \varnothing$	∞, b	$3, a$	∞, d	$1, a$	∞, f	∞, g
e		$5, e$	$2, e$	$2, e$	$1, a$	∞, f	∞, g
c		$5, e$	$2, e$	$2, e$		$3, c$	∞, g
d		$4, d$		$2, e$		$3, c$	$4, d$
b		$4, d$				$3, c$	$4, d$
f						$3, c$	$4, d$
$\boldsymbol{g}$							$4, d$

15 / 25

Illustration et trace de l'algorithme

Résultat :



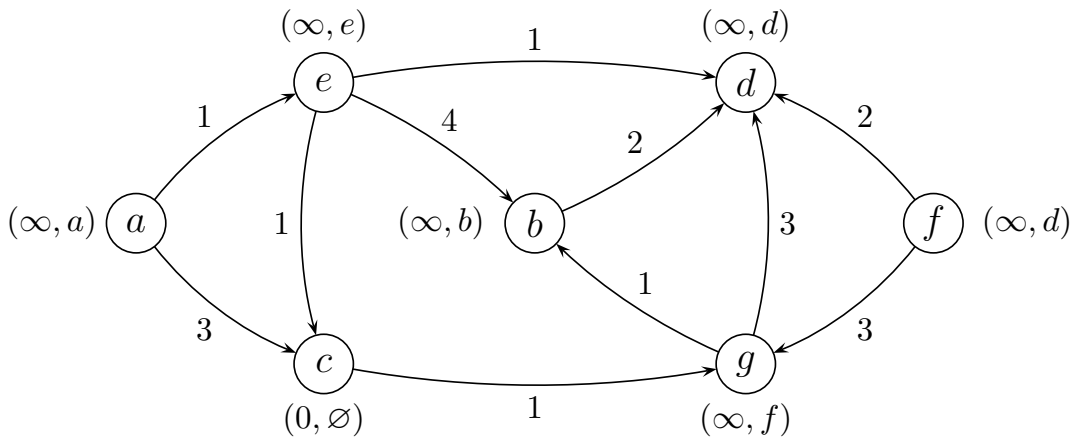
s	a	b	c	d	e	f	g
$\Pi[s]$	**0**	**4**	**2**	**2**	**1**	**3**	**4**
$\mathcal{A}[s]$	$\varnothing$	d	e	e	a	c	d

16 / 25

Cas où le sommet de départ n'est pas une source

Initialisations

$$r = c$$
$$L = (a, e, c, f, g, b, d)$$



ité.	a	b	c	d	e	f	g
init.	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g

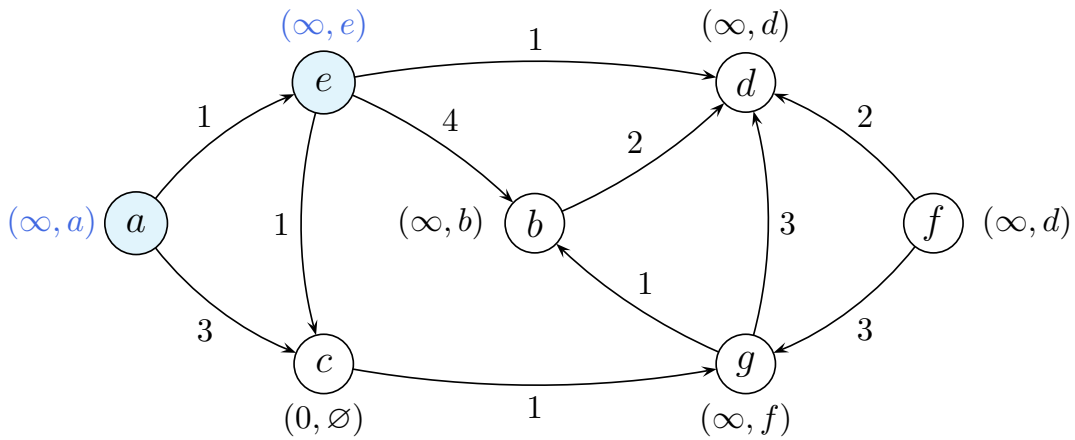
Rappel : les sommets sont examinés dans l'ordre topologique

17 / 25

Cas où le sommet de départ n'est pas une source

Itérations 1 et 2

$L = (\boldsymbol{a}, \boldsymbol{e}, c, f, g, b, d)$



ité.	a	b	c	d	e	f	g
init.	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
$\boldsymbol{a}, \boldsymbol{e}$	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g

Les sommets a et e sont à gauche de c dans L. Ils ne sont pas des descendants de c et ne seront pas dans la c-arborescence de cpm.

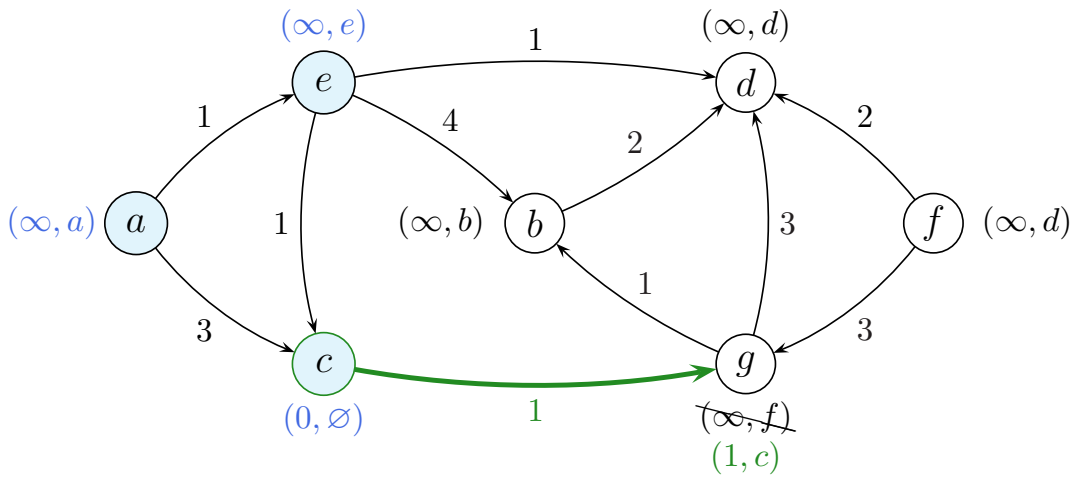
Les sommets a et e ont un potentiel infini, les arcs sortant de a et ceux sortant de e ne sont donc pas améliorants

18 / 25

Cas où le sommet de départ n'est pas une source

Itérations 3

$L = (a, e, \boldsymbol{c}, f, g, b, d)$



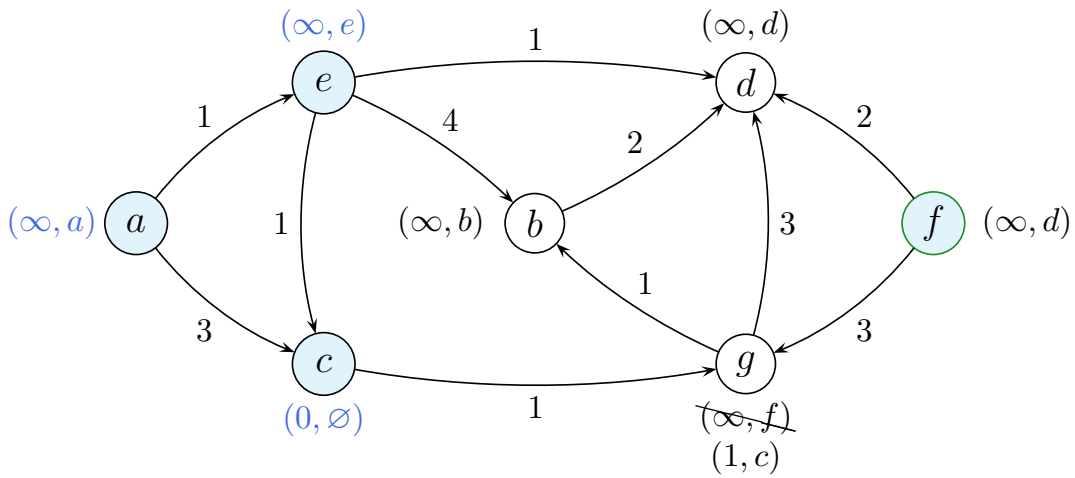
ité.	a	b	c	d	e	f	g
init.	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
a, e	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
$\boldsymbol{c}$		∞, b	$0, \varnothing$	∞, d		∞, f	$1, c$

19 / 25

Cas où le sommet de départ n'est pas une source

Itérations 4

$L = (a, e, c, \boldsymbol{f}, g, b, d)$



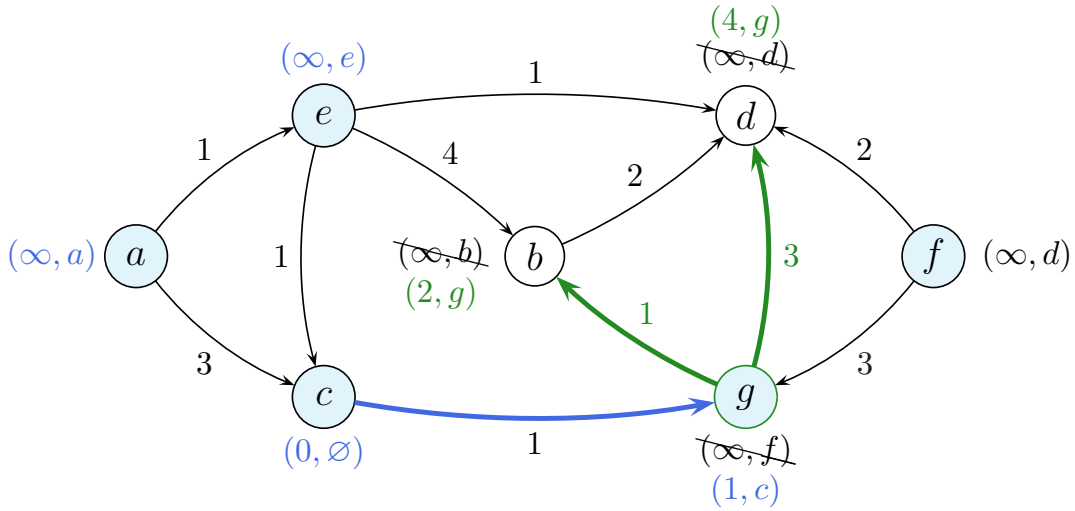
ité.	a	b	c	d	e	f	g
init.	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
a, e	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
c		∞, b	$0, \varnothing$	∞, d		∞, f	$1, c$
$\boldsymbol{f}$		∞, b		∞, d		∞, f	$1, c$

20 / 25

Cas où le sommet de départ n'est pas une source

Itérations 5

$L = (a, e, c, f, \boldsymbol{g}, b, d)$



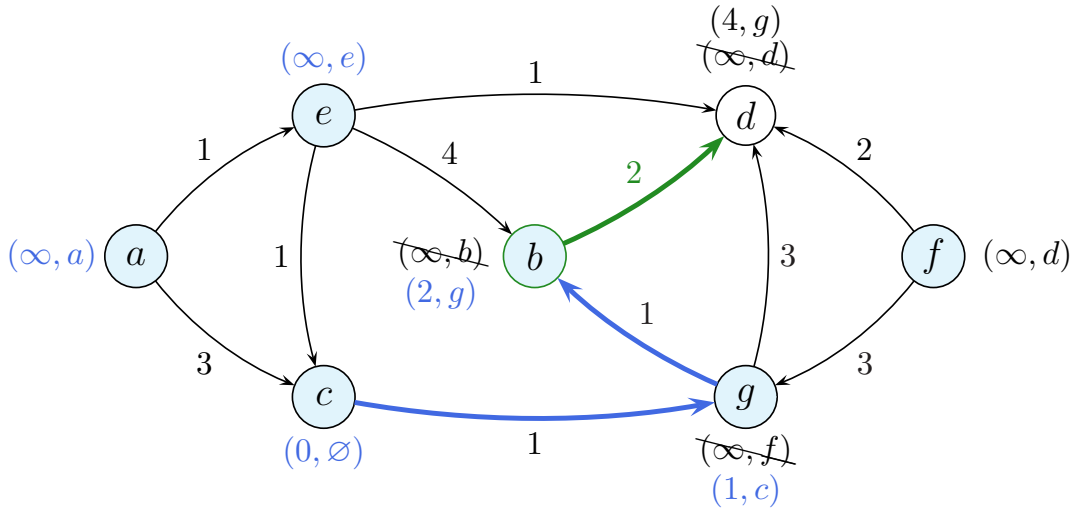
ité.	a	b	c	d	e	f	g
init.	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
a, e	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
c		∞, b	$0, \varnothing$	∞, d		∞, f	$1, c$
f		∞, b		∞, d		∞, f	$1, c$
$\boldsymbol{g}$		$2, g$		$4, g$			$1, c$

21 / 25

Cas où le sommet de départ n'est pas une source

Itérations 6

$L = (a, e, c, f, g, \boldsymbol{b}, d)$



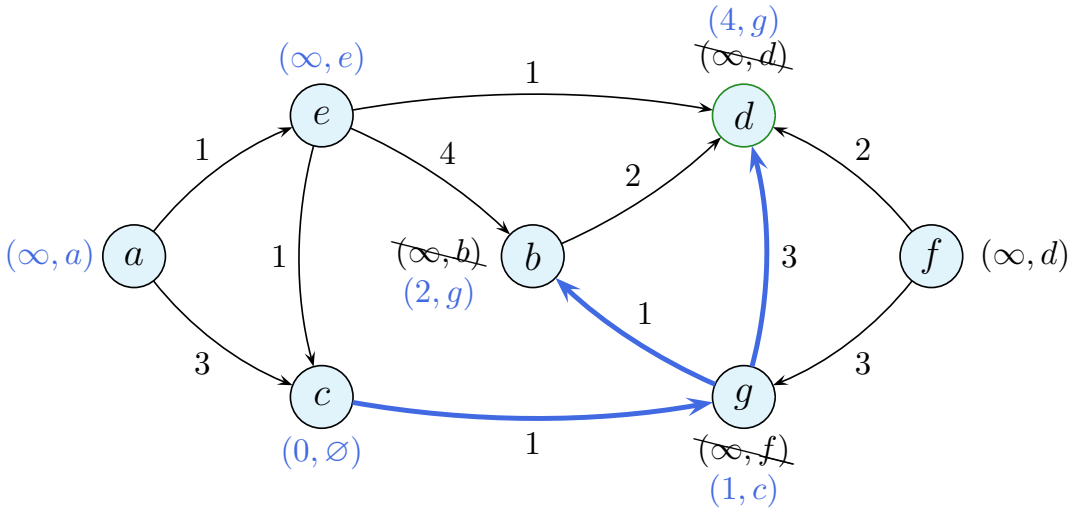
ité.	a	b	c	d	e	f	g
init.	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
a, e	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
c		∞, b	$0, \varnothing$	∞, d		∞, f	$1, c$
f		∞, b		∞, d		∞, f	$1, c$
g		$2, g$		$4, g$			$1, c$
$\boldsymbol{b}$		$2, g$		$4, g$			

22 / 25

Cas où le sommet de départ n'est pas une source

Itérations 7

$L = (a, e, c, f, g, b, \boldsymbol{d})$



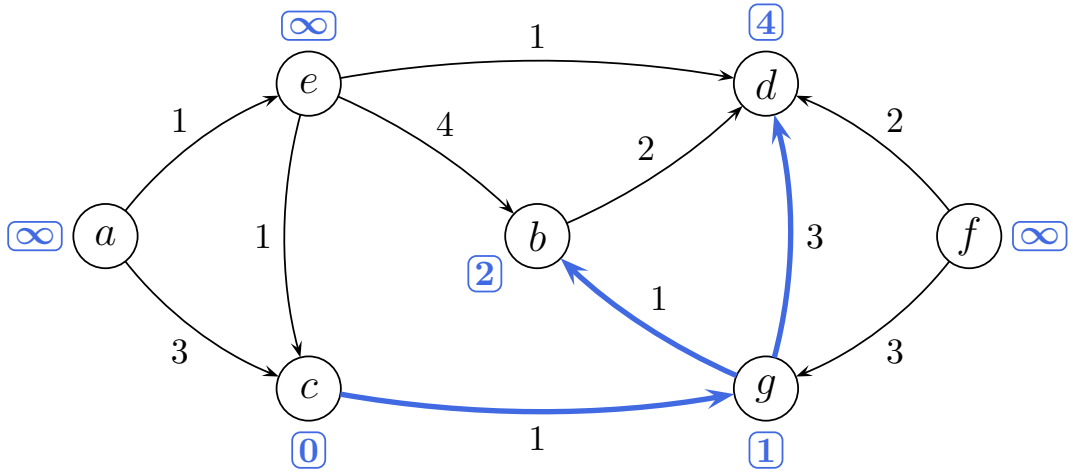
ordre topologique

ité.	a	b	c	d	e	f	g
init.	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
a, e	∞, a	∞, b	$0, \varnothing$	∞, d	∞, e	∞, f	∞, g
c		∞, b	$0, \varnothing$	∞, d		∞, f	$1, c$
f		∞, b		∞, d		∞, f	$1, c$
g		$2, g$		$4, g$			$1, c$
b		$2, g$		$4, g$			
$\boldsymbol{d}$				$4, g$			

23 / 25

Cas où le sommet de départ n'est pas une source

Résultat :



s	a	b	c	d	e	f	g
$\Pi[s]$	∞	**2**	**0**	**4**	∞	∞	**1**
$\mathcal{A}[s]$	a	g	$\varnothing$	g	e	f	c

24 / 25

Variantes possibles de l'algorithme

Recherche d'une r-arborescence de chemins de poids maximum.
(calcul des dates de début au plus tôt dans les méthodes PERT et MPM)

Recherche d'une r-arborescence de chemins de probabilité maximum. Quand les poids des arcs sont des probabilités, le poids d'un chemin est une probabilité égale au produit des poids des arcs du chemin.
(fiabilité dans des réseaux)

Recherche de tous les chemins de poids minimum (ou maximum) aboutissant en un sommet r. Le résultat est alors une anti-arborescence d'anti-racine r. Pour ce faire, il suffit d'examiner les sommets dans un ordre topologique inverse.
(calcul des dates de début au plus tard dans les méthodes PERT et MPM)

Remarque : l'algorithme de Bellman fonctionne en présence de poids négatifs (l'important est qu'il n'y ait pas de circuit).