Algorithme de Bellman-Ford

Manon Ruffini

Entrée : Un graphe orienté pondéré $G = (S, A, w)$ sans cycle de poids négatif[1] ; s l'origine de G
Sortie : Le poids d'un chemin de poids minimal ("plus court chemin") de s à t, pour tout sommet t de G

On note $S = [|1, n|]$, et on suppose que $s = 1$. De plus, pour tout sommet t de S, on note $\delta(t)$ le poids d'un plus court chemin de 1 à t (avec $\delta(t) = \infty$ si t n'est pas accessible depuis 1).

Étape 1

Définition des sous-problèmes

Pour $k \in \mathbb{N}$, on définit $\delta(t, k)$ comme le poids d'un chemin de poids minimal de 1 à t avec au plus k arcs. On remarque (principe des tiroirs) : $\delta(t, n-1) = \delta(t)$ (parce qu'il n'y a pas de cycle de poids négatif). On a les relations suivantes :

— $\delta(t, 0) = \begin{cases} 0 \text{ si } t = 1 \\ \infty \text{ sinon} \end{cases}$
— Pour tout entier k :

$$\delta(t, k) = \min\left\{\delta(t, k-1), \min_{u \in S | (u,t) \in A} \delta(u, k-1) + w(u, t)\right\}$$

Étape 2

On obtient l'algorithme suivant : 1 ; complexités

Algorithme 1 : Bellman-Ford-1(G)

$T[1 \dots n; 0 \dots n-1] \leftarrow$ matrice de taille $n \times n$, initialisée à ∞;
$T[1, 0] \leftarrow 0$;
pour $1 \leqslant k \leqslant n-1$ **faire**
  pour $t \in [|1, n|]$ **faire**
    $T[t, k] \leftarrow \min\left\{T[t, k-1], \min_{u \in S | (u,t) \in A} T[u, k-1] + w(u, t)\right\}$
  fin
fin
retourner $T[\cdot, n-1]$

Complexité spatiale : $O(n^2)$
Complexité temporelle : $O(n^2|A|)$

Étape 3

Optimisation de l'algorithme précédent

En fait, on peut même calculer en place les distances. De plus, quand on regarde, pour chaque sommet u, toutes les arêtes (u, a), pour trouver celles telles que $a = t$; on n'utilise vraiment qu'une seule fois chaque arête ; d'où l'algorithme 2.

1. Sans l'hypothèse d'absence de cycle de poids négatif : cf Compléments

Algorithme 2 : Bellman-Ford-2(G)

pour $t \neq 1$ **faire**
  $d[t] \leftarrow \infty$
fin
$d[1] \leftarrow 0$;
pour $1 \leqslant k \leqslant n-1$ `/* boucle (1)` `*/`
faire
  pour $(u,t) \in A$ `/* boucle(2)` `*/`
  faire
    $d[t] \leftarrow \min\{d[t], d[u] + w(u,t)\}$
  fin
fin
retourner d

Étape 4

Cet algorithme termine et est correct.

La terminaison est immédiate. Pour la correction, on va montrer un invariant de boucle.

Définition 1

On note $a_1 \dots a_r$ les arêtes de A, et on suppose que la boucle (2) parcourt les arêtes dans l'ordre a_1, puis a_2 ... jusqu'à a_r Pour tout $1 \leqslant k \leqslant n-1$, on définit :
- *d_k^0 = la valeur de d au début de la k-ème itération de la boucle (1)*
- *Pour tout $i \in [|1, r|]$, d_k^i = la valeur de d dans la k-ème itération de la boucle (1), après l'itération correspondant à l'arête a_i dans la boucle (2)*

Définition 2

On a l'invariant de boucle suivant :

$$\forall 0 \leqslant k \leqslant n-1, \forall t \in S, \forall 0 \leqslant i \leqslant r, \forall, \delta(t) \underset{(1)}{\leqslant} d_k^i[t] \underset{(2)}{\leqslant} \delta(t,k)$$

- Pour l'inégalité (1), il suffit de remarque que chaque $d_k^i[t]$ correspond au poids d'un chemin de 1 à t, avec au plus k arcs (ce qu'on peut montrer par récurrence).
- Pour l'inégalité (2), on procède par récurrence sur k :
  $k = 0$: C'est bon parce que $\delta(t,0) = \infty$ lorsque $t \neq 1$ et $\forall i, d_0^i[1] \leqslant d_0^0[1] = 0$
  Soit $k \in \mathbb{N}^*$: Supposons l'hypothèse vraie pour $k-1$, c'est-à-dire que

$$\forall t \in S, \forall 1 \leqslant i \leqslant r, d_{k-1}^i[t] \leqslant \delta(t, k-1)$$

  Alors :

$$\begin{aligned} d_k^0 &= \min\left\{ d_{k-1}^0[t], \min_{i \mid \exists u, a_i = (u,t) \in A} \left(d_{k-1}^{(u,t)}[u] + w(u,t) \right) \right\} \\ &\underset{\text{HR}}{\leqslant} \min\left\{ \delta(t,k-1), \min_{i \mid \exists u, a_i = (u,t) \in A} \delta(u,k-1) + w(u,t) \right\} \\ &= \delta(t,k) \end{aligned}$$

  Enfin, on remarque que : $\forall i \in [|1, r|], d_k^i[t] \leqslant d_k^0[t]$

Étape 5

Complexité spatiale : $O(|S|)$

Complexité temporelle : $O(|S||A|)$

Compléments

Détection de cycle de poids négatif

Si on ne suppose pas que G n'a pas de cycle de poids négatif, on peut modifier l'algorithme pour qu'il détecte les cycles de poids négatifs accessibles depuis 1. A la fin de l'algorithme, on ajoute :

Algorithme 3 : Bellman-Ford-2-suite

```
pour (u, t) ∈ A faire
    si d[u] + w(u, t) < d[t] alors
        retourner Il y a un cycle de poids négatif
fin
```

En effet, supposons qu'il existe un cycle de poids négatif $c = (u_0 \dots u_k)$, avec $u_0 = u_k$ accessible depuis 1. Alors, par définition :

$$\sum_{i=1}^{k} w(u_{i-1}, u_i) < 0$$

Par l'absurde, supposons que $\forall (u,t) \in A, d[u] + w(u,t) \geqslant d[t]$. Alors :

$$\forall 1 \leqslant i \leqslant k, d[u_{i-1}] + w(u_{i-1}, u_i) \geqslant d[u_i]$$

Donc :

$$\begin{aligned}\sum_{i=1}^{k} d[u_i] &\leqslant \sum_{i=1}^{k} (d[u_{i-1}] + w(u_{i+1}, u_i)) \\ &= \sum_{i=1}^{k} d[u_{i-1}] + \sum_{i=1}^{k} w(u_{i+1}, u_i) \\ &= \sum_{i=1}^{k} d[u_i] + \sum_{i=1}^{k} w(u_{i+1}, u_i) \text{ car } u_0 = u_k\end{aligned}$$

D'où :

$$0 \leqslant \sum_{i=1}^{k} w(u_{i+1}, u_i)$$

Contradiction.

Comment trouver un plus court chemin de 1 à t

Dans l'algorithme, il faut ajouter un tableau qui stocke le père de chaque sommet, qui est modifié dès qu'on trouve un poids de chemin plus faible (on ne stocke jamais le chemin tout entier, juste le père).

Références

[1] Thomas H. Cormen, *Algorithmique*. Dunod, 3è édition, 2010.