Rappel sur les algorithmes de plus court chemin

$G = (V, \mathcal{A}, w)$ un graphe dirigé avec une fonction de poids sur les arcs, $w : \mathcal{A} \to \mathbb{R}$.
On cherche à calculer le plus court chemin de tous les sources vers une destination fixée $i \in V$.

Dijkstra

Algorithme 1 : Dijkstra

Données : $G = (V, \mathcal{A}, w)$, $w \geq 0$;
Résultat : $d : V \to \mathbb{R}$ les longueurs des chemins les plus courts de tous les sommets vers i;
$\pi : V \to \mathbb{R}$ le premier sommet d'un chemin le plus court vers i ;
début
  $d(k) \leftarrow \begin{cases} 0 & \text{si } k = i \\ +\infty & \text{sinon} \end{cases}$ `/* initialiser avec un majorant */` ;
  $Q \leftarrow V$;
  tant que $Q \neq \emptyset$ **faire**
    $k \leftarrow \text{argmin}(d(j), j \in Q)$;
    $Q \leftarrow Q \backslash \{k\}$;
    pour chaque $j \in V$ *tel que* $(j, k) \in \mathcal{A}$ **faire**
      si $d(j) > d(k) + w(j, k)$ **alors**
        $\pi(j) \leftarrow k$;
        $d(j) \leftarrow d(k) + w(j, k)$;
      fin
    fin
  fin
fin

Implementation :
- avec un tableau : $O(|V|^2)$;
- avec un tas : $O((|\mathcal{A}| + |V|) \log |V|)$;
- avec un tas de Fibonacci : $O(|\mathcal{A}| + |V| \log |V|)$.

Bellman-Ford

Une autre solution est de procéder à toutes les relaxations possibles en parcourant l'ensemble des arcs à chaque étape.

Algorithme 2 : Bellman-Ford

Données : $G = (V, \mathcal{A}, w)$ qui n'admet pas de circuit de poids négatif ;
Résultat : $d : V \to \mathbb{R}$ les longueurs des chemins les plus courts de tous les sommets vers i;
$\pi : V \to \mathbb{R}$ le premier sommet d'un chemin le plus court vers i ;
début
  $d(k) \leftarrow \begin{cases} 0 & \text{si } k = i \\ +\infty & \text{sinon} \end{cases}$ `/* initialiser avec un majorant */` ;
  modified $\leftarrow 1$; iter $\leftarrow 1$;
  tant que *modified* $\neq 0$ *and iter* $\leq |V|$ **faire**
    modified $\leftarrow 0$, iter $\leftarrow$ iter $+ 1$;
    pour chaque $(k, j) \in E$ **faire**
      si $d(j) > d(k) + w(j, k)$ **alors**
        modified $\leftarrow 1$;
        $\pi(j) \leftarrow k$;
        $d(j) \leftarrow d(k) + w(j, k)$;
      fin
    fin
  fin
  si *modified* $\neq 0$ **alors retourner** *"il y a un cycle de poids négatif"*;
fin

Implementation : $O(|\mathcal{A}||V|)$.

Floyd-Warshall

Algorithme 3 : Floyd-Warshall

Données : $G = (V, \mathcal{A}, w)$ qui n'admet pas de circuit de poids négatif ; $V = \{1, \dots n\}$
Résultat : $d^n : V \times V \to \mathbb{R}$ les longueurs des chemins les plus courts pour tous les couples de sommets;
début
  $d^0(k, j) \leftarrow \begin{cases} w(k, j) & \text{si } (k, j) \in \mathcal{A} \\ +\infty & \text{sinon} \end{cases}$;
  pour $k = 1..n$ **faire**
    pour $i = 1..n$ **faire**
      pour $j = 1..n$ **faire**
        $d^k(i, j) \leftarrow \min\{d^{k-1}(i, j), d^{k-1}(i, k) + d^{k-1}(k, j)\}$;
      fin
    fin
  fin
fin

Implementation : $O(|V|^3)$.

Point commun de ces trois algorithmes est la relaxation :

$$d(j) \leftarrow \min_k \{d(j), d(k) + w(j, k)\}$$

ou

$$d(i, j) \leftarrow \min_k \{d(i, j), d(i, k) + d(k, j)\}$$