Algorithmes de plus courts chemins

Algorithme Dijkstra (**données** : graphe valué $G=(V, E, c)$ orienté avec $c(e)\geq 0$ $\forall e\in E$; sommet de départ s ; **résultat** : ensemble des potentiels π)

$\pi(s)=0$; $S=(s)$; $p=s$;
Pour tout $x\neq s$ **faire** $\pi(x)=+\infty$ **fin pour tout**
Tant que $V\neq S$ **faire**
    Pour tout $x\notin S$ tel que $e=(p, x)\in E$ **faire**
        Si $\pi(x) > \pi(p)+c(e)$ **alors** $\pi(x)= \pi(p)+c(e)$ **fin si**
    fin pour tout
    Choisir x tel que $\pi(x)=\min(\pi(y))$ $y\in V\text{-}S$
    $S=S\cup\{x\}$; $p=x$;
fin tant que
Retourner π
fin algorithme

Algorithme DijkstraV2 (**données** : graphe valué $G=(V, E, c)$ orienté avec $c(e)\geq 0$ $\forall e\in E$; sommet de départ s ; **résultat** : ensemble des potentiels π)

$\pi(s)=0$; $Q=(s)$; $p=s$;
Pour tout $x\neq s$ **faire** $\pi(x)=+\infty$ **fin pour tout**
Tant que $Q\neq\emptyset$ **faire**
    Choisir $p\in Q$ tel que $\pi(p)=\min(\pi(y))$ $y\in Q$
    $Q=Q\backslash\{p\}$
    Pour tout $x\in V$ tel que $e=(p, x)\in E$ **faire**
        Si $\pi(x) > \pi(p)+c(e)$ **alors**
            $\pi(x)= \pi(p)+c(e)$
            $Q=Q\cup\{x\}$
        fin si
    fin pour tout
fin tant que
Retourner π
fin algorithme

Algorithme A* (**données** : graphe valué $G=(V, E, c)$ orienté avec $c(e)\geq 0$ $\forall e\in E$; sommet de départ s, sommet destination d, heuristique $H(v)$ donnant pour tout sommet une borne inférieure à sa distance à la destination ; **résultat** : longueur du plus court chemin de s à d)

```
    π(s)=0 ; S=(s) ; p=s ;
    Pour tout x≠s faire π(x)=+∞ fin pour tout
    Tant que Q≠∅ faire
        Choisir p ∈ Q tel que π(p)=min(π(y)+H(y)) y∈Q
        Si p = d alors
            Retourner π(p)
        Q=Q\{p}
        Pour tout x∉Q tel que e=(p, x)∈E faire
            Si π(x) >π(p)+c(e) alors
                π(x)= π(p)+c(e)
                Q=Q∪{x}
            fin si
        fin pour tout
    fin tant que
    Retourner « Pas de chemin entre s et d »
fin algorithme
```

Algorithme Ford (**données** : graphe valué $G=(V, E, c)$ orienté ; sommet de départ s ; **résultat** : ensemble des potentiels π)

```
    π0(s)=0 ; k=0 ;
    Pour tout x≠s faire π0(x)=+∞ fin pour tout
    Faire
        k=k+1;
        πk(x)=min(πk−1(x), min((πk−1(y)+c(e), e=(y, x)∈E)
    Tant que ((k<n) et (∃x tel que (πk−1(x)≠(πk(x)))
    Si ((k=n) et (∃x tel que (πk−1(x)≠(πk(x))) alors il existe un circuit de longueur
négative
fin algorithme
```

Algorithme Bellman (**données** : graphe valué $G=(V, E, c)$ orienté sans circuit ; sommet de départ s ; **résultat** : ensemble des potentiels π ; Arborescence des plus courts chemins A)

```
    π(s)=0 ; S=(s) ;
    Tant que il existe x∉S dont tous les prédécesseurs y sont dans S faire
        π(x)=min( π(y)+c(e)), e=(y, x)∈E) ;
        Soit e*=(y*, x) un arc pour lequel π(x)=min( π(y)+c(e))
        A(x)= e* ; S=S∪{x}
    fin tant que
fin algorithme
```

`Algorithme` PERT (**`données`** : graphe valué $G=(V, E, c)$ orienté sans circuit ; sommets de début D et de fin F ; **`résultat`** : pour chaque tâche, la date au plus tôt π, la date au plus tard η et la marge δ)

$S=\{D\}$; $\pi(D)=0$;
`Tant que` il existe $x \notin S$ dont tous les prédécesseurs y sont dans S **`faire`**
$\pi(x)=\text{Max}(\ \pi(y)+c(e)),\ e=(y, x) \in E)$;
`fin tant que`
$S=\{F\}$; $\eta(F)=\pi(F)$
`Tant que` il existe $x \notin S$ dont tous les successeurs y sont dans S **`faire`**
$\eta(x)=\text{Min}(\ \eta(y)-c(e)),\ e=(x, y) \in E)$;
`fin tant que`
`Pour tout` $e=(x, y)$ **`faire`** $\delta(e)=\eta(y)-\pi(x)-c(e)$ **`fin pour tout`**
`fin algorithme`