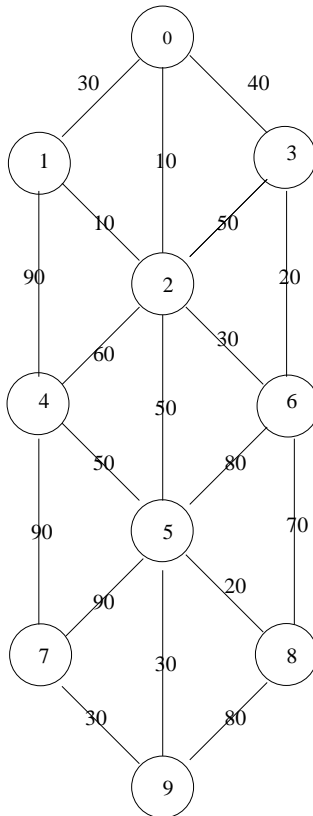


## TD d'Algorithmique 5

### Plus court chemin dans un graphe: Algorithme de Dijkstra

On considère un graphe possédant  $n$  sommets numérotés à partir de 0 et dont les arcs sont affectés de nombres entiers strictement positifs (leur *poids*), exprimant par exemple une distance. On note  $W(i, j)$  le poids de l'arc reliant les sommets  $i$  et  $j$ . Le fait qu'un tel arc n'existe pas se traduit par  $W(i, j) = +\infty$ .



Etant donné un sommet particulier (la *source*), l'algorithme de Dijkstra calcule, pour chaque sommet, un plus court chemin menant de la source au sommet considéré, ainsi que la distance correspondante.

On note  $S$  l'ensemble des sommets dont la plus courte distance à la source est connue, et  $Q$  les autres. On note aussi  $dist(i)$  la plus courte distance de  $i$  à la source quand on passe uniquement par des sommets de  $S$  et  $pred(i)$  désigne le prédécesseur de  $i$  dans un plus court chemin.  $dist$  et  $pred$  sont initialisés comme suit:

```
INITIALISATION
Pour tout sommet  $i$  faire
     $dist(i) = +\infty$ 
     $pred(i) = -1$ 
 $dist(source) = 0$ 
```

L'algorithme de Dijkstra est alors le suivant:

```
DIJKSTRA
INITIALISATION
 $S = \emptyset$ 
 $Q = \{0, \dots, n-1\}$ 
Tant que  $Q \neq \emptyset$  faire
     $s = \text{EXTRACT-MIN}(Q)$ 
     $S = S \cup \{s\}$ 
    Pour tout sommet adjacent à  $s$  faire  $\text{RELAX}(s,t)$ 
```

Dans cet algorithme  $\text{EXTRACT-MIN}(Q)$  ôte de  $Q$  le sommet  $s$  tel que  $dist(s)$  est minimale et renvoie ce sommet  $s$ . L'algorithme  $\text{RELAX}$  est le suivant:

```
RELAX( $s,t$ )
if  $dist(t) > dist(s) + W(s,t)$  alors
     $dist(t) = dist(s) + W(s,t)$ 
     $pred(t) = s$ 
```

1. Implanter cet algorithme en Java en utilisant des matrices d'adjacence et le faire tourner sur le graphe donné en illustration. On trouvera ci-dessous la trace souhaitée.
2. Démontrer la correction de cet algorithme.

```
0: distance = 0    chemin : 0
1: distance = 20   chemin : 0 2 1
2: distance = 10   chemin : 0 2
3: distance = 40   chemin : 0 3
4: distance = 70   chemin : 0 2 4
5: distance = 60   chemin : 0 2 5
6: distance = 40   chemin : 0 2 6
7: distance = 120  chemin : 0 2 5 9 7
8: distance = 80   chemin : 0 2 5 8
9: distance = 90   chemin : 0 2 5 9
```