

# Algorithmique des Graphes

## L3 informatique

13 janvier 2020

Vous êtes invités à remettre une copie claire, concise, sans rature ni surcharge. Il est par ailleurs inutile de recopier l'énoncé. . . La note finale tiendra compte de la présentation générale de la copie.

**Q1.** Quelle est la version orientée de l'affirmation : un cycle élémentaire passe par autant de sommets que d'arêtes. Les affirmations correspondantes sont elles correctes?

**Q2.** Soit  $p > 0$  un entier. On considère un graphe non orienté d'ordre  $2p$  dont chaque sommet est de degré supérieur ou égal à  $p$ . Peut-on affirmer que la longueur d'un plus court chemin entre deux sommets est inférieure ou égale à  $p$  ?

Dans graphe  $G(S, A)$  pondéré par une fonction  $w: A \rightarrow \mathbb{Z}$  à valeurs dans l'ensemble des entiers relatifs, on définit le coût du chemin  $\mu = [x_0, x_1, \dots, x_r]$  par

$$w(\mu) = \sum_{i=1}^r w(x_{i-1}x_i)$$

Un chemin minimal de  $x$  à  $y$  est un chemin de coût minimal d'origine  $x$  et d'extrémité  $y$ . Le coût d'un chemin minimal est noté  $\delta(x, y)$ . On pose  $\delta(x, y) = +\infty$  s'il n'existe pas de chemin de  $x$  vers  $y$ .

**Q3.** Donner un exemple de graphe d'ordre 3 ayant deux sommets  $x$  et  $y$  tel que :

$$\delta(x, y) = -\infty \quad \text{et} \quad \delta(y, x) = +\infty$$

**Q4.** Montrer que pour tous sommets  $x$  et  $y$  :

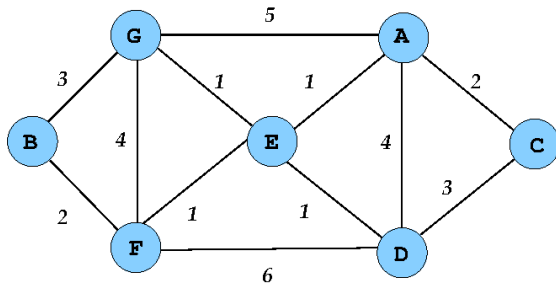
$$\delta(s, y) \leq \delta(s, x) + w(x, y).$$

L'algorithme de Dijkstra détermine le coût des chemins minimaux d'une origine  $s$  aux autres sommets d'un graphe pondéré par une fonction **positive**. Il utilise deux sous-ensembles  $E$  et  $F$  des sommets du graphe. Une estimation du coût minimal d'un chemin entre  $s$  et  $x$  est maintenue dans  $d[x]$ . À chaque itération un des sommets  $u \in F$  minimisant  $d$  et qui n'est pas dans  $E$  est sélectionné (ligne 13), il est retiré de  $F$ , ajouté à  $E$ , et les estimations des voisins sont mises à jour par la procédure relâchement.

```
DIJKSTRA( G, w, s )
E := ensemble vide
F := sommets de G
d := [+inf, +inf, ..., +inf]
d [s] := 0
tantque non vide ( F )
  u := extraire( E, F )
  retirer u de F
  ajouter u a E
  pour chaque voisin v de u
    relacher( u, v, w)
```

```
1 relacher( u, v, w )
2 si d [ v ] > d [ u ] + w(u, v)
3   alors
4     d [ v ] := d [ u ] + w(u, v)
```

**Q5.** Quel algorithme du cours est similaire à l'algorithme de Dijkstra ?



**Q6.** Faire tourner l'algorithme de Dijkstra sur le graphe ci-dessus en prenant le sommet  $A$  pour origine. Tracer le tableau  $d$  des estimations après chaque itération.

**Q7.** L'algorithme de Dijkstra s'arrête. Pourquoi ?

**Q8.** Il est possible de représenter les ensembles  $E$  et  $F$  par un unique tableau de booléens. Pourquoi ? Montrer qu'avec ce seul point de vue, le coût des extractions est au plus quadratique en l'ordre du graphe.

**Q9.** On note  $n$  l'ordre du graphe. On suppose une implantation naïve dans laquelle le graphe est représenté par une matrice d'adjacence. Formuler le temps de calcul en fonction de  $n$ .

**Q10.** Préciser les structures optimales pour représenter l'ensemble  $F$ , les arcs du graphes afin d'accélérer l'extraction de  $u$  et le parcours des voisins. Formuler le temps de calcul de Dijkstra correspondant en fonction du nombre de sommets  $n$  et du nombre d'arcs  $m$ .

**Q11.** Modifier l'algorithme pour écrire une procédure `print(t)` qui, appelée en ligne 19, afficherait un chemin minimal d'origine  $s$  d'extrémité  $t$ .

**Q12.** Un circuit absorbant est un circuit de coût strictement négatif. Il n'est pas toujours possible de définir le chemin de coût minimal en présence d'un circuit absorbant. Pourquoi ?

**Q13.** Donner un exemple de graphe pondéré par une fonction à valeurs dans  $\mathbb{Z}$ , sans circuit absorbant, et sur lequel l'algorithme de Dijkstra échoue.

```

1 BELLMAN-FORD( G, w, s )
2 d := [ inf, inf, ..., inf ]
3 d [ s ] := 0
4
5 repeter ordre(G) fois
6   pour chaque arc uv
7     relacher(u, v, w)
8
9 pour chaque arc uv
10  si d[v] > d[u] + w(u,v) alors
11    retourner FAUX
12 retourner VRAI

```

L'algorithme de Bellman-Ford utilise la procédure de relâchement de l'algorithme de Dijkstra. Il donne les plus courts chemin d'origine unique à condition qu'aucun circuit absorbant ne soit accessible à partir de l'origine.

**Q14.** Estimer le temps de calcul de l'algorithme de Bellman-Ford en fonction du nombre de sommets  $n$  et du nombre d'arcs  $m$ .

**Q15.** On considère des graphes à  $n$  sommets et  $m := n\sqrt{n}$  arcs. Une implantation de Bellman-Ford traite une instance à 1000 sommets en 1 seconde. Estimer le temps de calcul pour une instance d'ordre  $10^6$ .



*Edsger Dijkstra, 11/05/1930–6/08/2002, informaticien néerlandais, prix Turing 1972 (The Humble Programmer), renommé pour, entre bien d'autres choses, ses travaux sur les systèmes d'exploitation et le célèbre algorithme qui porte son nom.*