université de BORDEAUX

Année 2022-2023 — DS novembre 2023
**Épreuve de :** Recherche Opérationnelle
**Date :** 24 novembre 2023 — **Durée :** 1h30
**Épreuve de :** O. Baudon — **Nb. pages : 7**

---

*Une attention toute particulière sera portée à la présentation et à la rédaction de la copie.*

---

Dans tous les exercices, on désignera par $V(G)$ et $E(G)$ respectivement l'ensemble des sommets et l'ensemble des arêtes d'un graphe $G$. Les variables $n$ et $m$ désigneront respectivement le nombre de sommets et d'arêtes.

Dans tout le sujet, on conviendra que, dans les différentes structures de données modélisant les graphes, les sommets sont rangés dans l'ordre croissant de leur nom.

# 1 Structures de données et complexité

Dans un graphe simple orienté $G$, on appelle ***puits universel*** tout sommet $v$ tel que :
— pour tout sommet $u \neq v$, il existe un arc $uv$ ;
— $v$ n'a pas de successeur.

**1.1)** Dessiner un graphe à 4 sommets possédant un puits universel. Quel est le nombre maximum de puits universels que peut contenir un graphe ? (Justifier votre réponse).

**1.2)** Pour chacune des deux représentations :

1. listes de successeurs,
2. matrice d'adjacence,

écrire un algorithme pour déterminer si le graphe contient un puits universel et en étudier la complexité. *Attention de présenter les algorithmes de façon lisible !*

# 2 Flot Maximum

**2.1)** En utilisant l'algorithme de Ford-Fulkerson rappelé ci-dessous (Algorithme 1), trouver le flot maximum entre les sommets $s$ et $t$ du réseau de la figure 1. Le flot possède déjà une valeur initiale donnée par le premier nombre porté sur les arcs, le second étant la capacité de l'arc. Par exemple, l'arc `(s,A)` possède un flot initial de 2 et une capacité de 4. On donnera pour chaque exécution de la procédure de marquage le chemin augmentant obtenu et l'augmentation correspondante.

**2.2)** Donner la coupe minimum obtenue par la dernière exécution de la procédure de marquage, sa valeur et redessiner le graphe avec le flot maximum (en précisant sur un arc retour sa valeur).

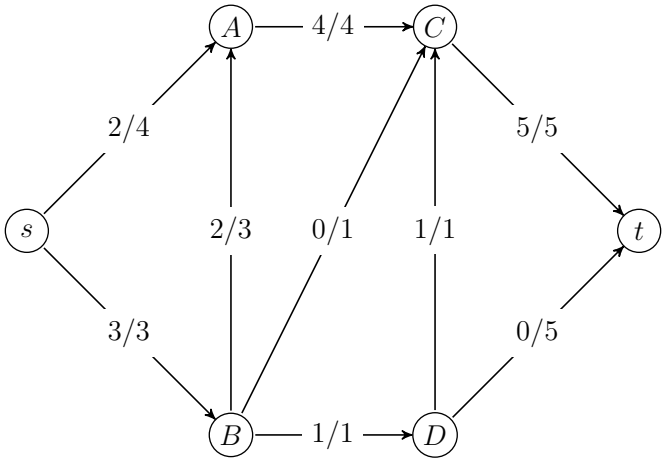


FIGURE 1 – Réseau

Dans l'Algorithme 1 (FlotMax), le paramètre $c$ désigne les capacités du graphe $G$, $s$ la source et $t$ la destination du flot $f$ calculé. $I(e)$ et $T(e)$ désignent respectivement le sommet initial et le sommet terminal d'un arc $e$.

**Algorithme 1** FlotMax$(G, c, s, t)$

```
1: pour tout e de E(G) faire
2:    f[e] ← 0
3: fin pour
4: répéter
5:    Marquage(G, c, f, s, t)
6:    si t ∈ Y alors
7:       v ← t
8:       C⁺ ← {(t, s)}
9:       C⁻ ← ∅
10:      tant que v ≠ s faire
11:         e ← A[v]
12:         si v = T[e] alors
13:            C⁺ ← C⁺ ∪ {e}
14:            v ← I[e]
15:         sinon
16:            C⁻ ← C⁻ ∪ {e}
17:            v ← T[e]
18:         fin si
19:      fin tant que
20:   fin si
21:   pour tout e ∈ C⁺ faire
22:      f(e) ← f(e) + δ[t]
23:   fin pour
24:   pour tout e ∈ C⁻ faire
25:      f(e) ← f(e) − δ[t]
26:   fin pour
27: jusqu'à t ∉ Y
```

**Algorithme 2** Marquage$(G, c, f, s, t)$

1: $Y \leftarrow \{s\}$
2: $\delta(s) \leftarrow +\infty$
3: $Max \leftarrow$ faux
4: **tant que** $t \notin Y et Max =$faux **faire**
5:  **si** il existe $e = (u, v)$ avec $u \in Y, v \notin Y, f(e) < c(e)$ **alors**
6:  $Y \leftarrow Y \cup \{v\}$
7:  $A[v] \leftarrow e$
8:  $\delta[v] \leftarrow min(\delta[u], c(e) - f(e))$
9:  **sinon**
10:  **si** il existe $e = (u, v)$ avec $v \in Y, u \notin Y, f(e) > 0$ **alors**
11:  $Y \leftarrow Y \cup \{u\}$
12:  $A[u] \leftarrow e$
13:  $\delta[u] \leftarrow min(\delta[v], f(e))$
14:  **sinon**
15:  Max $\leftarrow vrai$
16:  **fin si**
17:  **fin si**
18: **fin tant que**

# 3 Plus courts chemins

Un étudiant du Master Miage de l'Université de Bordeaux, désirant faire un séjour linguistique, décide de se rendre en Suède. Après avoir fait le tour de quelques compagnies, il a recensé plusieurs connexions aériennes lui permettant d'aller de Bordeaux à Stockholm. Il les a représentées à l'aide du graphe suivant (Figure 2) :

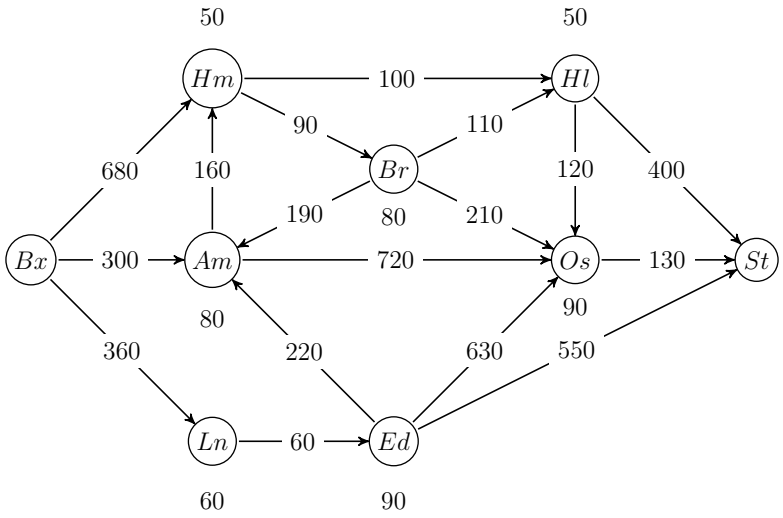


FIGURE 2 – Voyage

où Bx = Bordeaux, Hm = Hambourg, Am = Amsterdam, Ln = Londres, Ed = Edimbourg, Br = Berlin, Hl = Helsinki, Os = Oslo, St = Stockholm.

Cependant, les horaires des vols sont tels que l'étudiant est obligé de passer la nuit dans chacune des villes où il fait escale.

Les valeurs sur les arcs correspondent aux prix en euros pour les vols, et les valeurs à coté des sommets représentent le prix en euros à payer pour passer la nuit dans un hôtel de la ville correspondante.

**3.1)** On désire trouver une solution optimale pour ce problème. Quelles modifications faut-il appliquer au graphe pour pouvoir utiliser l'algorithme de Dijkstra ?

**3.2)** Calculer la solution optimale. On donnera en particulier la suite des valeurs prises par la variable `pivot` et pour chacune de ces valeurs, les modifications des valeurs du tableau `d` associées.
Dessiner l'arborescence des plus courtes distances dans le graphe modifié.

**Algorithme 3** Dijkstra$(G, w, s)$

```
1: pour tout v de V(G) faire
2:    d(v) ← ∞
3:    pere(v) ← NIL
4:    couleur(v) ← BLANC
5: fin pour
6: d(s) ← 0
7: F ← FILE_PRIORITE({s}, d)
8: tant que F ≠ ∅ faire
9:    pivot ← EXTRAIRE_MIN(F)
10:   pour tout e = (pivot, v) arc sortant de pivot faire
11:      si couleur(v) = BLANC alors
12:         si d(v) = ∞ alors
13:            INSERER(F, v)
14:         fin si
15:         si d[v] > d[pivot] + w(e) alors
16:            d[v] ← d[pivot] + w(e)
17:            pere[v] ← pivot
18:         fin si
19:      fin si
20:   fin pour
21:   couleur[pivot] ← NOIR
22: fin tant que
```

# 4 Arbre de poids maximum

On considère le problème suivant :

un réseau routier reliant un ensemble de villes est représenté par un graphe $G$ simple, non orienté, chaque ville étant représentée par un sommet et deux sommets $a$ et $b$ étant reliés par une arête s'il existe une route directe reliant la ville $a$ à la ville $b$.

Chaque arête $e$ est munie d'un poids $h(e)$ donnant la hauteur maximum (en cm) d'un véhicule pouvant emprunter cette route. On souhaite calculer les itinéraires entre les villes maximisant la hauteur possible d'un véhicule.

On considère $T_{max}$ l'arbre couvrant de $G$ de poids maximum.

**4.1)** Montrer que si une arête $e = \{u, v\}$ n'appartient pas à $T_{max}$, et si $C_{uv}$ désigne la chaîne reliant $u$ et $v$ dans $T_{max}$, alors $\forall e' \in C_{uv}, h(e) \leq h(e')$.

**4.2)** Que pouvez-vous en conclure si les poids sont tous différents ?

**4.3)** Que faut-il modifier à l'algorithme de Prim, rappelé ci-dessous (Algorithme 4), pour calculer un arbre de poids maximum ?

**4.4)** Appliquer l'algorithme de Prim modifié au réseau routier de la Figure 3, en partant du sommet $A$. Donner l'ensemble des routes à emprunter pour maximiser la hauteur des camions utilisables pour tout transport entre deux villes, et préciser dans quel ordre les sommets ont été rajoutés à l'arbre obtenu.

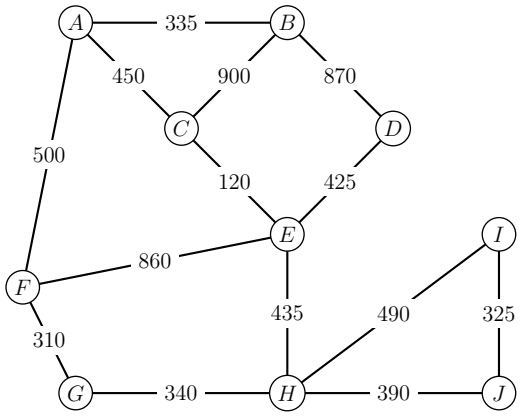


FIGURE 3 – Réseau routier

**Algorithme 4** Prim$(G, w)$

```
1: F ← FILE_PRIORITE(V(G), cle)
2: pour tout v de V(G) faire
3:    cle(v) ← ∞
4: fin pour
5: Soit r un sommet de V(G)
6: cle(r) ← 0
7: pere(r) ← NIL
8: tant que F ≠ ∅ faire
9:    u ← EXTRAIRE_MIN(F)
10:   pour tout v ∈ Adj(u) faire
11:      si v ∈ F et w(u, v) < cle(v) alors
12:         pere(v) ← u
13:         cle(v) ← w(u, v)
14:      fin si
15:   fin pour
16: fin tant que
```