université de BORDEAUX

Année 2021-2022 — DS novembre 2021
Épreuve de : Recherche Opérationnelle
Date : 12 novembre 2021 — **Durée :** 1h30
Épreuve de : O. Baudon — **Nb. pages : 5**

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.

Les algorithmes utilisés dans les exercices sont disponibles à la fin du sujet.

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 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 1) :

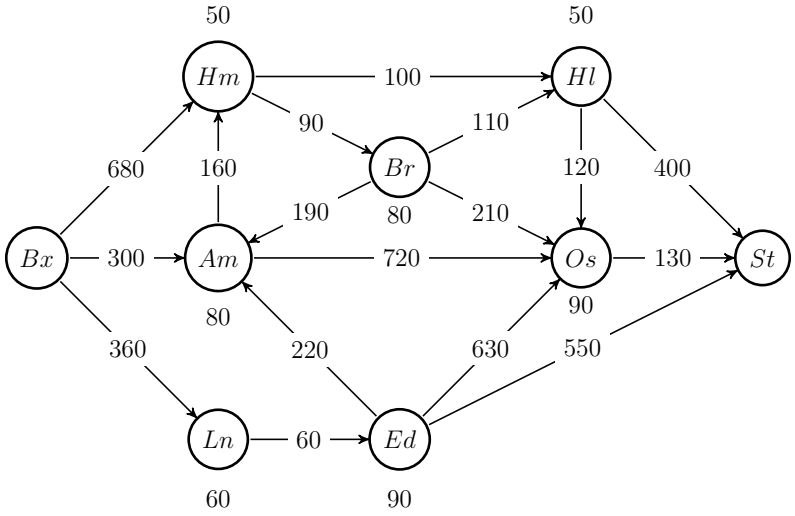

FIGURE 1 – Voyage

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.

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

2.2) 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.

3 Application du parcours en profondeur

L'objectif de cet exercice est de fournir un algorithme de recherche d'un chemin hamiltonien dans un tournoi.

Un *tournoi* est un graphe simple complet orienté. Autrement dit, si T est un tournoi et u et v deux sommets de T, alors $(u, v) \in E(T) \Leftrightarrow (v, u) \notin E(T)$.

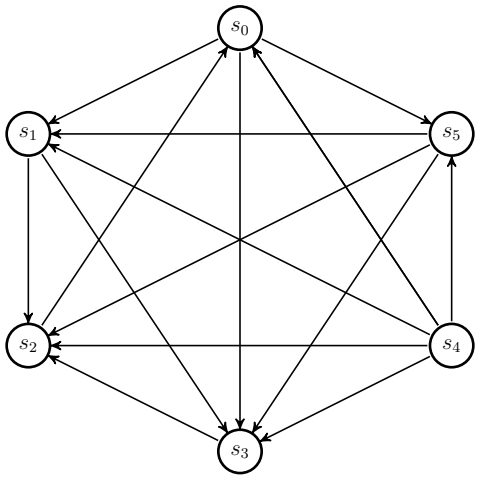

FIGURE 2 – Exemple de tournoi à 6 sommets

Un *chemin hamiltonien* est un chemin qui passe par tous les sommets du graphe une et une seule fois.
Théorème : tout tournoi possède un chemin hamitonien. Par exemple, dans le tournoi de la Figure 2 : $s_4, s_3, s_2, s_0, s_5, s_1$ est un chemin hamiltonien.

3.1) Appliquer l'algorithme de parcours en profondeur `PP` au tournoi T de la Figure 2. On conviendra que dans les listes d'adjacence, les sommets sont rangés dans l'ordre des indices et également que la boucle principale de `PP` (lignes 5 à 7) considère les sommets dans l'ordre des indices. Pour chaque sommet, donner les valeurs `d`, `f`, `pere` retournées par l'algorithme en respectant l'ordre de visite des sommets. Ordonner les sommets de T dans l'ordre inverse de leur fin de visite. Que remarquez-vous ?

On rappelle que les arcs obtenus lors d'un parcours en profondeur peuvent être de 4 types :

- arcs de liaison : entre un sommet et ses fils dans l'arborescence ;
- arcs de retour (ou arrière) : entre un sommet et l'un de ses ancêtres ($\neq$ de son père) dans l'arborescence ;
- arcs avant : entre un sommet et l'un de ses descendants ($\neq$ de ses fils) dans l'arborescence ;
- arcs transverses : entre deux sommets sans relation dans l'arborescence.

3.2) Lorsque l'on range les sommets dans l'ordre inverse de leur fin de visite, quels sont les types d'arcs reliant deux sommets consécutifs ? Montrer que si les sommets sont rangés dans l'ordre inverse de leur fin de visite, alors si le sommet v suit immédiatement le sommet u dans cet ordre, l'arc entre ces deux sommets est orienté de u vers v.

3.3) Modifier l'algorithme de parcours en profondeur pour afficher les sommets dans un ordre induisant un chemin hamiltonien. Quelle est la complexité de l'algorithme obtenu ?

3.4) Démontrer le théorème donné en début d'exercice : tout tournoi possède un chemin hamiltonien. *Indication : montrer qu'un chemin de longueur maximale contient tous les sommets.*

4 Rappels

```
0   Dijkstra(G,w,s)
1     pour chaque sommet v de V(G) faire
2         d[v] <- infini
3         pere[v] <- nil
4         couleur[v] <- BLANC
5     d[s] <- 0
6     F <- FILE_PRIORITE(V(G), d)
7     tant que F non vide faire
8         pivot <- EXTRAIRE_MIN(F)
9         couleur[pivot] <- GRIS
10        pour tout arc e = (pivot, v) arc sortant de pivot faire
11            si couleur(v) = BLANC et d[v] > d[pivot] + w(e)
12                alors d[v]  <- d[pivot] + w(e)
13                      pere[v]  <- pivot
14        couleur[pivot] <- NOIR
```

```
0   PP(G)
1     pour chaque sommet v de V(G) faire
2         couleur[v] <- BLANC
3         pere[v] <- nil
4     temps <- 0
5     pour chaque sommet v de V(G) faire
6         si couleur[v] = BLANC
7             alors Visiter_PP(v)
```

```
0   Visiter_PP(u)
1     couleur[u] <- GRIS
2     d[u] <- temps <- temps + 1
3     pour chaque v de Adj[u] faire
4         si couleur[v] = BLANC
5             alors pere[v] <- u
6                   Visiter_PP(v)
7     couleur[u] <- NOIR
8     f[u] <- temps <- temps + 1
```