Aix*Marseille université

Domaine Sciences et Technologies
Portail 1 René Descartes

Science informatique : TD 4

Année 2023–24

Plus court chemin, cycles eulériens, coloration

Exercice 1 (Plus court chemin) Lorsque les graphes sont équipés de poids (sur les arcs), il peut être intéressant de calculer des plus courts chemins entre des paires de sommets. On décrit un graphe (orienté) pondéré à l'aide d'une matrice d'adjacence M avec autant de lignes et de colonnes qu'il y a de sommets dans le graphe, et dont le coefficient pour u et v deux sommets vaut

$$M_{u,v} = \begin{cases} +\infty & \text{s'il n'y a pas d'arcs de } u \text{ à } v \\ \text{poids de } u \to v & \text{sinon} \end{cases}$$

Le poids d'un chemin $u_0 \to u_1 \to u_2 \to \cdots \to u_{n-1} \to u_n$ du graphe (où $u_0 \to u_1$, $u_1 \to u_2$, …, $u_{n-1} \to u_n$ sont des arcs du graphe) est égal à la somme

$$M_{u_0,u_1} + M_{u_1,u_2} + \cdots + M_{u_{n-1},u_n} = \sum_{i=0}^{n-1} M_{u_i,u_{i+1}}$$

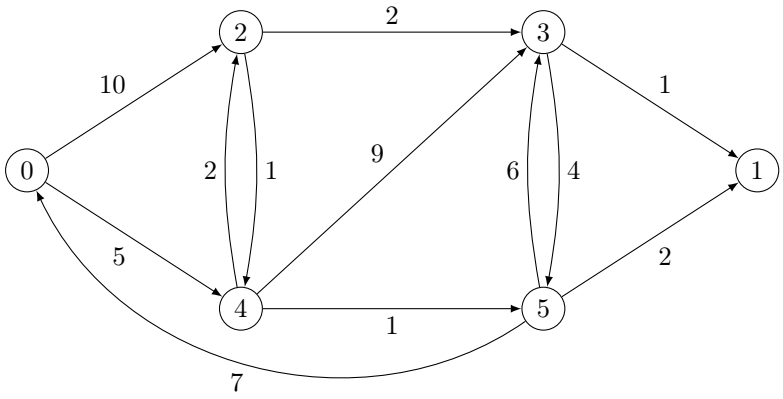
Un plus court chemin de u à v est un chemin menant de u à v dont le poids est minimal parmi tous les chemins possibles.

Voici une description de l'algorithme de Dijkstra :

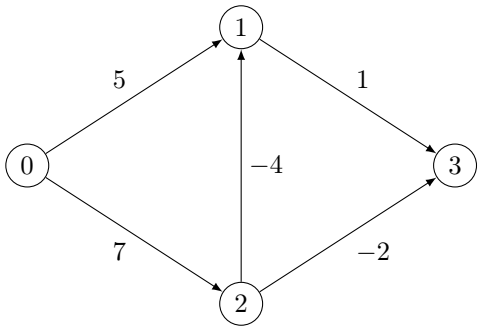
- colorier le sommet de départ en rouge et les autres sommets en blanc
- attribuer au sommet de départ la distance provisoire 0
- attribuer aux autres sommets la distance provisoire $+\infty$
- attribuer à chaque sommet le prédécesseur nul
- créer une file de priorité (dont la priorité est la distance provisoire)
- insérer le sommet de départ dans la file
- tant que la file n'est pas vide :
  - extraire de la file le sommet u ayant la distance minimale
  - pour chaque sommet v adjacent à u :
    - si v est blanc alors colorier v en rouge et enfiler v dans la file
    - si la distance de u + le poids de (u, v) est inférieur à la distance de v alors colorier v en rouge, mettre u comme prédécesseur de v et attribuer à v la distance de u + le poids de l'arc (u, v)
- renvoyer la liste des prédécesseurs

La différence notable entre le parcours en largeur et l'algorithme de Dijkstra réside dans l'utilisation d'une file de priorité qui nécessite de maintenir les estimations des distances et de mettre à jour les arcs rouges : pour cela, plutôt que de maintenir un graphe `H` comme précédemment, on associe à chaque sommet v rouge son prédécesseur dans le graphe `H`, c'est-à-dire l'unique sommet u tel que (u, v) est un arc rouge de `H`. Initialement, on choisit d'attribuer à tout sommet 0 comme prédécesseur.

1. Exécuter l'algorithme de Dijkstra sur l'exemple ci-dessous en partant de la source $s = 0$:



2. En déduire un plus court chemin du sommet 0 au sommet 1.
3. Jusque-là, nous avons étudié uniquement des graphes pondérés avec des poids entiers positifs ou nuls : pourtant, on pourrait imaginer des graphes avec des poids négatifs, par exemple si le poids de l'arc représente des échanges d'argent (vente ou achat de produits). Exécuter l'algorithme de Dijkstra sur l'exemple ci-dessous où plusieurs arcs ont des poids négatifs :



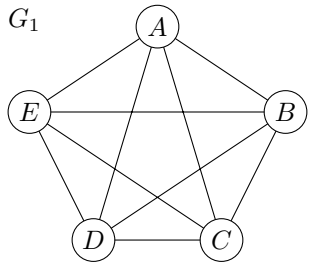
4. Qu'en déduisez-vous sur l'algorithme de Dijkstra ?

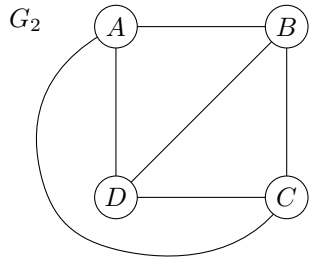
Exercice 2 (Cycles eulériens)

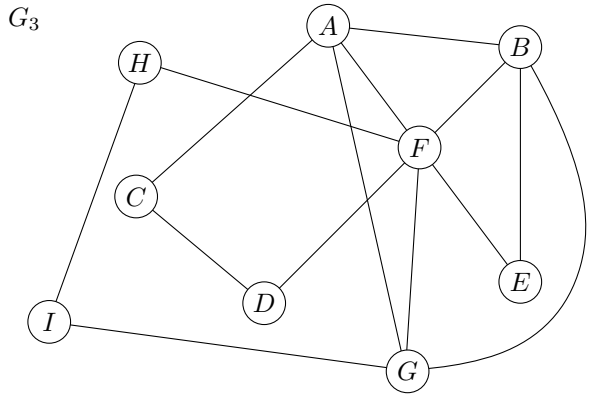
Dans un graphe non orienté, on appelle *cycle eulérien* tout cycle (un chemin qui revient à son point de départ) qui traverse chaque arête du graphe une et une seule fois. La caractérisation suivante existe :

> « Un graphe non orienté connexe admet un cycle eulérien
> si et seulement si chaque sommet est de degré pair. »

1. En utilisant la caractérisation précédente, dites pour chacun des graphes suivants s'ils admettent un cycle eulérien.





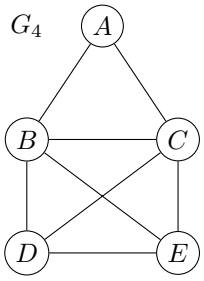


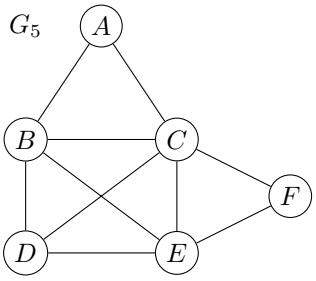
L'algorithme de Hierholzer, décrit de manière informelle ci-dessous, permet de construire un cycle eulérien (s'il en existe un) :

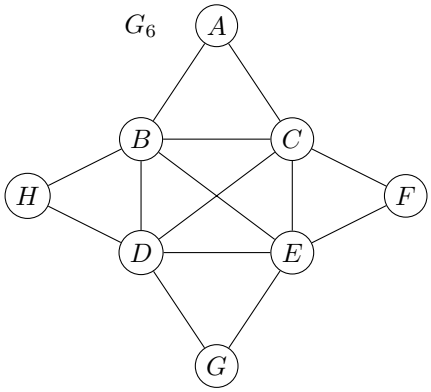
- Choisir n'importe quel sommet initial v
- Suivre un chemin arbitraire d'arêtes jusqu'à retourner à v, obtenant ainsi un cycle c
- **Tant qu**'il y a des sommets u dans le cycle c avec des arêtes qu'on n'a pas encore choisies **faire**
  - Suivre un chemin à partir de u, n'utilisant que des arêtes pas encore choisies, jusqu'à retourner à u, obtenant un cycle c'
  - Prolonger le cycle c par c'

À noter qu'en toute généralité, un graphe eulérien peut admettre plusieurs cycles eulériens différents.

2. Pour les graphes précédents qui admettent un cycle eulérien, trouver un tel cycle en appliquant l'algorithme de Hierholzer.
3. La notion de cycle eulérien peut être étendue : un *chemin eulérien* est un chemin (pas nécessairement un cycle) qui traverse chaque arête du graphe une et une seule fois. Parmi les graphes suivants, quels sont ceux pour lesquels vous pouvez trouver un chemin eulérien ?







4. À partir de vos observations, conjecturer une caractérisation des chemins eulériens en terme de degré des sommets du graphe (ressemblant à la caractérisation des cycles eulériens).

Exercice 3 (Coloration de graphes)

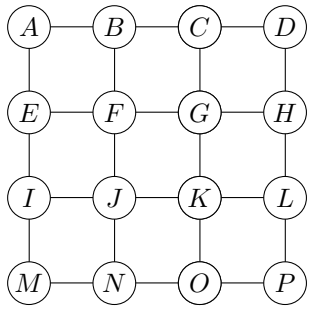
L'algorithme de Welsh-Powell, décrit informellement ci-dessous, permet de colorer les sommets d'un graphe de manière que deux sommets adjacents soient de couleurs différentes. On choisit dans cet exercice de colorer les sommets avec des couleurs qui sont des entiers $0, 1, 2, \ldots$.

- Trier le sommets du graphe par ordre de degré décroissant
- couleur $\leftarrow 0$ (*couleur initiale*)
- **Tant qu**'il y a encore des sommets non colorés **faire**
  - Parcourir la liste triée des sommets et colorer en *couleur* les sommets non colorés qui ne sont pas connectés à d'autres sommets de la même couleur
  - couleur $\leftarrow$ couleur $+ 1$ (*choisir une nouvelle couleur*)

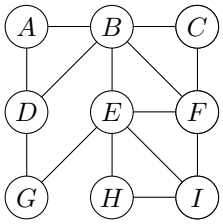
À chaque fois que deux sommets ont le même degré, *on les suppose triés par ordre alphabétique.* Par exemple, si les sommets A, D, B sont tous de degré 3, on suppose qu'ils seront triés dans l'ordre A, B, D.

1. Colorier les graphes suivants à l'aide de l'algorithme de Welsh-Powell.

G_1



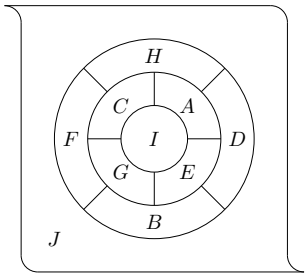
G_2



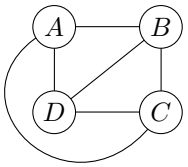
2. Les colorations obtenues sont-elle optimales en terme du nombre de couleurs utilisées ? Si oui, pourquoi ? Si non, trouver une meilleure coloration.

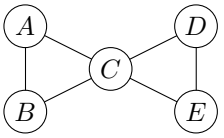
Exercice 4 (Coloration de cartes)

Une carte est un découpage d'une feuille de papier en régions, séparées par des frontières. Par exemple, la carte à droite est découpée en 10 régions. Chaque carte est associée à un *graphe planaire* (un graphe qu'on peut dessiner sur une feuille de papier sans qu'aucune des arêtes n'en croise une autre). Vice versa, chaque graphe planaire est associé à une carte (non nécessairement unique). Chaque région de la carte correspond à un sommet et chaque frontière entre régions à une arête.



1. Dessiner une carte pour chacun des graphes suivants.





2. À l'aide de l'algorithme de Welsh-Powell, colorier la carte donnée en exemple au début de l'exercice. *Noter que la région externe J correspond, elle aussi, à l'un des sommets du graphe associé. (Comme dans l'exercice précédent, si deux sommets ont le même degré, choisir d'abord le plus petit selon l'ordre alphabétique.)*
3. La coloration obtenue est-elle optimale en terme de nombre de couleurs ? Si oui, pourquoi ? Si non, trouver une meilleure coloration.