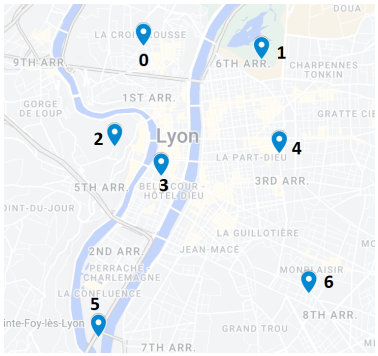
I Algorithme de Christofides

On considère un graphe $G = (V, E)$ **non orienté**, **complet** et **pondéré** par une fonction $w : E \longrightarrow \mathbb{R}^+$. On supposera dans toute la suite que $V = \{0, ..., n-1\}$ où n est le nombre de sommets. Si G' est un graphe ou un ensemble d'arêtes, son **poids** $w(G')$ est la somme des poids des arêtes de G'.

On utilisera comme exemple le graphe G_{ex} suivant, représenté par matrice d'adjacence pondérée, dont les sommets sont des points d'intérêts de Lyon (numérotés de 0 à 6) et chaque arête correspond à une distance euclidienne entre les deux points :



$$G_{ex} = \begin{pmatrix} 0.0 & 2.3 & 1.75 & 2.2 & 2.9 & 4.8 & 5.1 \\ 2.3 & 0.0 & 2.9 & 2.6 & 1.5 & 5.4 & 4.0 \\ 1.75 & 2.9 & 0.0 & 0.9 & 2.9 & 3.2 & 4.2 \\ 2.2 & 2.6 & 0.9 & 0.0 & 2.1 & 2.9 & 3.2 \\ 2.9 & 1.5 & 2.9 & 2.1 & 0.0 & 4.4 & 2.5 \\ 4.8 & 5.4 & 3.2 & 2.9 & 4.4 & 0.0 & 3.7 \\ 5.1 & 4.0 & 4.2 & 3.2 & 2.5 & 3.7 & 0.0 \end{pmatrix}$$

On suppose que G est **métrique**, c'est-à-dire que w vérifie l'inégalité triangulaire : $\forall x, y, z \in V, \quad w(x, y) \leq w(x, z) + w(z, y)$

Un **cycle hamiltonien** dans un graphe est un cycle qui passe exactement une fois par chaque sommet. Dans la suite, on représente un cycle hamiltonien par sa liste de sommets, dans l'ordre (contenant donc chaque sommet exactement une fois).

Le **problème du voyageur de commerce (TSP)** consiste à trouver un cycle hamiltonien de poids minimum.

L'**algorithme de Christofides** donne une $\frac{3}{2}$-approximation du TSP. Voici son fonctionnement :

- Trouver un arbre couvrant de poids minimum T de G.
- Considérer V_1 l'ensemble des sommets de degré impair de T.
- Trouver un couplage parfait de poids minimum M de $G[V_1]$.
- Considérer le multigraphe H obtenu par union des arêtes de T et de M.
- Trouver un circuit eulérien C dans H.
- Transformer C en cycle hamiltonien en « sautant » les sommets répétés.

I.1 Recherche d'un arbre couvrant de poids minimum

1. Quel algorithme peut-on utiliser pour trouver un arbre couvrant de poids minimum dans un graphe ? L'appliquer sur G_{ex} en donnant les arêtes d'un arbre couvrant de poids minimum T_{ex} obtenu par l'algorithme, dans l'ordre.

I.2 Sommets de degré impair

Soit V_1 l'ensemble des sommets de degré impair dans T (en ne considérant que les arêtes de T).

2. Donner V_1 dans le cas où $T = T_{ex}$.
3. Montrer qu'un graphe G' non orienté possède un nombre pair de sommets de degré impair.

La question précédente montre donc que V_1 est de cardinal pair.

I.3 Couplage parfait de poids minimum

On définit $G[V_1]$ (**graphe induit** par V_1) comme le graphe dont l'ensemble de sommets est V_1 et dont deux sommets sont adjacents si et seulement si ils sont adjacents dans G. Les poids des arêtes de $G[V_1]$ sont les mêmes que ceux de G.

Un **couplage parfait de poids minimum** est un couplage parfait dont le poids est minimum parmi tous les couplages parfaits possibles.

4. Montrer l'existance d'un couplage parfait de poids minimum de $G[V_1]$.

5. On suppose que les sommets de G sont des points de $\mathbb{R}^2$ (comme c'est le cas pour G_{ex}). On peut alors considérer chaque arête comme un segment dans $\mathbb{R}^2$. Montrer que si M est un couplage de G dont deux arêtes se croisent (c'est-à-dire que les segments correspondants s'intersectent), alors M n'est pas de poids minimum.

6. Avec $G = G_{ex}$ et $T = T_{ex}$, donner les arêtes d'un couplage parfait de poids minimum M_{ex} de $G[V_1]$, ainsi que son poids.

I.4 Cycle eulérien

Un **multigraphe** est comme un graphe, sauf qu'il peut y avoir plusieurs arêtes entre deux sommets.
On définit le multigraphe $H = (V, E_H)$ dont les sommets sont les mêmes que G et tel que E_H contienne l'union des arêtes de T et de M (avec répétition : E_H est un multiensemble).

7. Dessiner H dans le cas où $G = G_{ex}$, $T = T_{ex}$ et $M = M_{ex}$.

Dans un graphe G' non orienté, un **cycle eulérien** est un cycle passant par chaque arête exactement une fois (mais qui peut passer plusieurs fois par un sommet, contrairement à un cycle hamiltonien).

8. Donner un cycle eulérien dans H dans le cas où $G = G_{ex}$, $T = T_{ex}$ et $M = M_{ex}$.

9. Montrer que si G' possède un cycle eulérien C, alors G' est connexe et tous les sommets de G' sont de degré pair.

On veut écrire un algorithme en OCaml permettant de trouver un cycle eulérien dans un graphe.

10. Écrire une fonction `supprime e l` supprimant la première occurence de `e` dans la liste `l`. Si `e` n'apparaît pas dans `l`, on renverra `l` inchangée. Par exemple, `supprime 3 [6; 3; 2; 3; 5]` doit renvoyer `[6; 2; 3; 5]`.

Dans la suite, on suppose que G' est connexe et que tous les sommets de G' sont de degré pair.

11. On part d'un sommet u quelconque de G', puis, tant que possible, on se déplace suivant une arête adjacente à u qui n'a pas encore été utilisée. Montrer que ce processus termine et que la liste des sommets visités forme un cycle C.

12. Écrire une fonction `cycle g u` qui renvoie le cycle C obtenu par l'algorithme précédent sous forme d'une liste de sommets, où G' est représenté par une liste d'adjacence `g`. On supprimera de `g` les arêtes utilisées par le cycle.

13. Montrer que G' possède un cycle eulérien, en raisonnant par récurrence.

14. En déduire une fonction `euler g u` qui renvoie un cycle eulérien (sous forme d'une liste de sommets) en partant depuis le sommet `u` dans le graphe G' donné sous forme de liste d'adjacence.

I.5 Cycle hamiltonien

Le cycle eulérien C dans le graphe H de la section précédente peut passer plusieurs fois par le même sommet. Pour le transformer en cycle hamiltonien, on supprime les sommets apparaissant plusieurs fois dans C (à part la première occurence).

15. Donner le cycle hamiltonien obtenu avec $G = G_{ex}$ ainsi que son poids. Comparer avec le cycle hamiltonien de poids minimum.

16. Écrire une fonction `supprime_doublons l` qui renvoie la liste obtenue en supprimant les doublons dans la liste de sommets `l`, sauf la première occurence de chaque sommet.
Par exemple, `supprime_doublons [1; 0; 1; 4; 6; 4]` doit renvoyer `[1; 0; 4; 6]`.

I.6 Preuve de la $\frac{3}{2}$-approximation

Soit C^* un cycle hamiltonien de poids minimum de G et T un arbre couvrant de poids minimum de G.

17. Montrer que $w(T) \leq w(C^*)$.

Soit V_1 l'ensemble des sommets de degré impair de T. Soit C_1^* un cycle hamiltonien de poids minimum de $G[V_1]$.

18. Montrer qu'on peut séparer les arêtes de C_1^* en deux couplages parfaits M_1 et M_2 de $G[V_1]$.

19. Soit M un couplage parfait de poids minimum de $G[V_1]$. Montrer que $w(M) \leq \min(w(M_1), w(M_2))$.

20. Montrer que $\min(w(M_1), w(M_2)) \leq \dfrac{w(C_1^*)}{2}$.

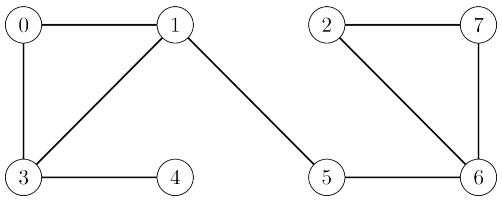
21. Montrer que $w(C_1^*) \leq w(C^*)$.

22. Soit C le cycle hamiltonien obtenu par l'algorithme de Christofides. Montrer que $w(C) \leq \dfrac{3}{2} w(C^*)$.

II Pont et algorithme de Tarjan

Soit $G = (V, E)$ un graphe connexe non orienté. Un **pont** de G est une arête telle que $G - e$ n'est pas connexe.

1. Donner un algorithme en complexité quadratique pour trouver tous les ponts d'un graphe G.
2. Donner tous les ponts du graphe ci-dessous.



Soit $r \in V$. Soit T une arborescence de parcours en profondeur depuis r.

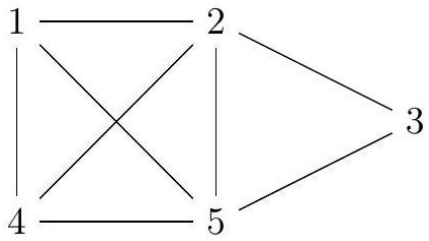
On définit un graphe orienté $\vec{G}$ en orientant les arcs de T dans le sens du parcours et les arcs arrières en direction de l'ancêtre. On définit $p(v)$ comme étant la profondeur du sommet v dans T. On définit $l(v)$ comme étant la plus petite valeur de $p(u)$ pour un sommet u accessible (par un chemin) depuis v dans $\vec{G}$.

3. On effectue un parcours en profondeur depuis 0 dans le graphe de la question 2, en visitant à chaque fois le plus petit sommet en priorité lorsqu'il y a plusieurs choix. Dessiner $\vec{G}$ et les valeurs de $p(v)$ et $l(v)$ pour chaque sommet v.
4. À quelle condition sur l un arc (u, v) de T est-il un pont ?
5. Donner une relation de récurrence permettant de calculer $l(v)$.
6. En déduire un algorithme en OCaml permettant de trouver tous les ponts d'un graphe G. On choisira des structures de données adaptées. Quelle est la complexité de cet algorithme ?

III Calcul des triangles

On considère un graphe non-orienté $G = (V, E)$ où $V = \{1, \ldots, n\}$ est l'ensemble des sommets et E est un ensemble d'arêtes qui sont des paires de sommets de V. Un triangle dans G est un sous-ensemble $\{x, y, z\} \in V$ tel que $\{x, y\}, \{y, z\}$, et $\{x, z\}$ appartiennent à E. Dans ce problème, on veut concevoir des algorithmes qui prennent en entrée un graphe et calculent (sans doublons) l'ensemble des triangles du graphe.

1. Déterminer les triangles du graphe suivant:



2. Si G est représenté par matrice d'adjacence, proposer un algorithme naïf en $O\left(|V|^3\right)$ pour déterminer l'ensemble des triangles de G.

Dans la suite du sujet, on supposera toujours que le graphe d'entrée est fourni sous forme de listes d'adjacence, et on supposera toujours que chacune de ces listes est triée.

3. Soient l_1 et l_2 deux listes triées. Proposer un algorithme en complexité linéaire pour calculer l'intersection de ces deux listes (les éléments appartenant à la fois à l_1 et l_2).
4. Soit Δ le degré maximal d'un sommet de G. Proposer un algorithme en temps $O(|E| \times \Delta)$ pour déterminer l'ensemble des triangles de G. Commenter la performance de cet algorithme.
5. Si l'on compare l'algorithme de la question 2 et celui de la question 4, lequel a la meilleure complexité? Le choix de la représentation du graphe d'entrée était-il important?
6. Un sommet lourd est un sommet de degré $\geq \sqrt{E}$. Montrer que G contient au plus $2\sqrt{E}$ sommets lourds.
7. Proposer un algorithme en $O\left(|E|^{3/2} + |V|\right)$ pour déterminer l'ensemble des triangles de G.

On suppose dans la suite que G n'a pas de triangle et on veut majorer le nombre d'arêtes de G.

8. Montrer que pour toute arête $e = \{u, v\}$ de G, $\deg(u) + \deg(v) \leq |V|$.
9. En déduire que $\sum_{v \in V} \deg(v)^2 \leq |V||E|$.
10. En déduire que $|E| \leq \dfrac{|V|^2}{4}$ (théorème de Mantel).
11. Donner un exemple de graphe sans triangle et vérifiant $|E| = \dfrac{|V|^2}{4}$.

IV Questions sur les arbres couvrants

Soit $G = (V, E)$ un graphe pondéré.

1. Soit C un cycle de G et $e = \{u, v\}$ une arête de C dont le poids est strictement supérieur au poids des autres arêtes de C. Montrer que e ne peut pas appartenir à un arbre couvrant de de poids minimum de G.
2. (Propriété d'échange) Soient T_1, T_2 deux arbres couvrants de G et e_2 une arête de $T_2 - T_1$. Montrer qu'il existe une arête e_1 de T_1 telle que $T_1 - e_1 + e_2$ (le graphe obtenu en remplaçant e_1 par e_2 dans T_1) est un arbre couvrant de G.
3. Montrer que si tous les poids des arêtes de G sont différents, alors G admet un unique arbre couvrant de poids minimum.
4. Soit T_1 un arbre couvrant de poids minimum de G et T_2 le 2ème plus petit arbre couvrant, c'est-à-dire l'arbre couvrant de poids minimum en excluant T_1. Montrer que T_1 et T_2 diffèrent d'une arête et en déduire un algorithme pour trouver T_2.