LINMA1691: Théorie des graphes
Devoir 3 : Parcours eulériens

1 Contexte

Sacha du Bourg-Palette est un dresseur de Pokémon de la région de Kanto où vit le célèbre professeur Chen. Celui-ci lui a demandé de compléter le Pokédex en capturant tout les Pokémon de la région en visitant chacune des routes de Kanto. Sacha étant d'un naturel pressé et têtu comme une mule, il refuse d'emprunter deux fois la même route. En utilisant vos compétences en théorie des graphes et en Python, aidez Sacha dans sa quête épique en lui donnant un itinéraire qui lui convienne.

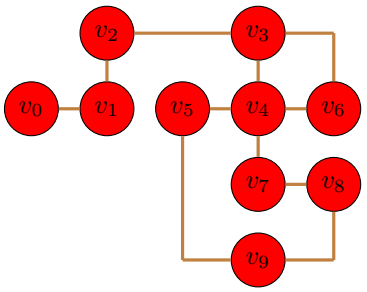

FIGURE 1 – Région de Kanto sous forme de graphe (légèrement modifiée).

2 Énoncé

Pour ce devoir, vous devez coder une fonction `eulerian_path_finder` qui prend en entrée un graphe simple et retourne en sortie un parcours eulérien si celui existe et `None` si un tel parcours n'existe pas. La complexité attendue est $\mathcal{O}(E+V)$ (linéaire relativement au nombre d'arêtes et de noeuds) [1].

2.1 Input de la fonction

Le graphe est donné en input sous forme de liste d'adjacence. Une arête incidente aux noeuds v_j et v_k sera donc représentée deux fois. La liste d'adjacence suivante est la représentation du graphe 1 :

```
[[1],[0, 2],[1, 3],[2, 4, 6],[3, 5, 7, 6],[4, 9],[3, 4],[4, 8],[7, 9],[5, 8]]
```
(1)

2.2 Output de la fonction

La réponse attendue doit être représentée par une liste des indices des noeuds du parcours emprunté qu'il soit ouvert ou fermé. Attention : le graphe donné en entrée ne doit PAS être modifié. La fonction `deepcopy`

1. Pour parvenir à cette complexité, l'utilisation de `set` peut être utile.

du module standard `copy` pourrait vous être utile pour cette partie. Si le parcours trouvé est fermé, le noeud de départ et de fin doit être indiqué deux fois. Pour le graphe 1, une solution possible est :

```
[0, 1, 2, 3, 6, 4, 7, 8, 9, 5, 4, 3]
```
(2)

Comme précisé précédemment, s'il s'avère que le graphe ne possède aucun parcours eulérien, la fonction devrait répondre `None`. Pour vérifier rapidement vos solutions vous pouvez vérifier que la longueur de votre liste en sortie soit égale au nombre d'arête plus un.

2.3 Jeux de tests

Pour vous aider, quelques graphes vous sont donnés dans des fichiers textes [2]. Dans chacun des fichier, la première ligne annonce le nombre de noeuds et chacune des lignes suivantes donne la liste d'adjacence du noeud correspondant. Une fonction de lecture des fichier test donnant un input utilisable vous est également fourni. Les solutions n'étant pas uniques pour un graphe donné, il ne tient qu'à vous de vérifier les réponses de votre fonction.

3 Soumission

Vous devez soumettre votre fichier solution "`pathfinder_hw.py`" complété sur l'activité Inginious [3] du même nom.(Note : Pas de texte "print" dans le fichier.).

Le langage de programmation est **Python 3** (version 3.5).

- **Échéance :** vendredi 1er décembre 2023 à 22h.
- **Questions :** Contactez `thibaud.schoenauen@student.uclouvain.be` ***et*** `brieuc.pinon@uclouvain.be` (en copie).

2. Attention, les graphes peuvent être différents de ceux des tests Inginious.
3. `https://inginious.info.ucl.ac.be`