Rapport de Projet

1. Introduction

Le projet consiste à analyser des collaborations entre acteurs en utilisant des concepts de théorie des graphes. Le jeu de données, fourni au format JSON, contient des informations sur les collaborations, que nous devons convertir en un graphe exploitable avec Networkx. Ensuite, différentes requêtes sont implémentées pour répondre à des questions spécifiques sur les collaborations et les distances entre les acteurs.

2. Répartition des Tâches

Les tâches ont été réparties comme suit :

- **Recherche sur les algorithmes** : Réalisée par Romain et Ambroise.
- **Conversion JSON vers Graphe** : Réalisée par Romain et Ambroise.
- **Implémentation des fonctions** :
    - Romain : `collaborateurs_communs`, `est_proche`, `dist_naive`, `distance`.
    - Ambroise : `centralite`, `centre_hollywood`, `eloignement_max`, `centralite_groupe`.
- **Documentation et rapport** : Collaboration entre Romain et Ambroise.

Tâche	Romain	Ambroise
Recherche sur les algorithmes	X	X
Conversion JSON vers Graphe	X	X
Collaborateurs en Commun	X	
Calcul des Distances	X	
Centralité d'un Acteur		X
Calcul du Diamètre du Graphe		X
Centre d'un Groupe d'Acteurs		X
Documentation et Rapport	X	X

Question 6.2 : Collaborateurs en commun

Comment exprimeriez-vous cette notion (ensemble des collaborateurs en commun) en termes de théorie des graphes ? Pouvez-vous donner une borne inférieure sur le temps nécessaire à l'exécution de votre fonction ?

Réponse : En théorie des graphes, les collaborateurs en commun de deux acteurs sont les voisins communs aux deux nœuds correspondants. La complexité minimale est O(d(u)+d(v))O(d(u) + d(v))O(d(u)+d(v)), où d(u)d(u)d(u) et d(v)d(v)d(v) sont les degrés des nœuds u et v.

Question 6.3 : Collaborateurs proches

Reconnaissez-vous l'algorithme classique en théorie des graphes qui est au cœur de ce programme ? Grâce à la fonction précédente, comment pouvez-vous déterminer si un acteur se trouve à distance k d'un autre acteur ? Tentons maintenant de déterminer la distance entre deux acteurs. Est-ce que ré-utiliser la fonction précédente vous semble intéressant ? Donnez la complexité (asymptotique) d'un tel algorithme. Comment pouvez-vous maintenant modifier la fonction qui vous a été fournie afin de trouver la distance entre deux acteurs ? Donnez la complexité d'un tel algorithme.

Réponse :

1. **Algorithme classique** : L'algorithme utilisé est le Breadth-First Search (BFS).
2. **Distance k** : On peut utiliser BFS jusqu'à une profondeur de k.
3. **Réutilisation de la fonction** : Réutiliser la fonction précédente n'est pas efficace pour trouver la distance exacte. Une simple modification de BFS pour retourner la profondeur lorsqu'un acteur est trouvé est suffisante. La complexité est O(V+E) pour chaque exécution de BFS.

Question 6.4 : Qui est au centre d'Hollywood ?

On cherche maintenant à déterminer la centralité d'un acteur. Pour cela, on veut déterminer la plus grande distance qui le sépare d'un autre acteur dans le graphe. Quelle notion de théorie des graphes permet de modéliser cela ? Proposez une fonction qui calcule la centralité d'un acteur dans Gc. A l'aide de la fonction précédente, écrivez une autre fonction qui va déterminer l'acteur le plus central du graphe Gc.

Réponse :

1. **Notion de théorie des graphes** : La notion est l'excentricité d'un nœud.
2. **Calcul de la centralité** : Voir la fonction `centralite`.
3. **Acteur le plus central** : Voir la fonction `centre_hollywood`.

Question 6.5 : Une petite famille

Nous allons maintenant tenter de déterminer si deux acteurs peuvent être très éloignés dans Gc. Plus précisément, vous fournirez une fonction permettant de déterminer la distance maximale dans Gc entre toute paire d'acteurs/actrices. Est-ce que ce nombre est bien inférieur ou égal à 6 pour le jeu de données fourni ?

Réponse : La distance maximale dans un graphe est appelée le diamètre. Nous utilisons la fonction `eloignement_max` pour calculer cette distance.

Question 6.6 : Bonus

Proposez une méthode similaire à celle calculant le centre du graphe mais qui se restreint ici à déterminer le centre d'un groupe d'acteurs (On remarquera qu'un tel sommet ne fait pas

nécessairement partie du groupe en question). Faites en sorte, quand cela a du sens, que vos fonctions renvoient un sous-graphe de Gc plutôt qu'une simple liste d'acteurs.

Réponse : Nous utilisons la fonction `centralite_groupe` pour déterminer le centre d'un groupe d'acteurs.

4. Évaluation Expérimentale

- **Mesure de Performance** :
  - **Métriques Utilisées** : Temps d'exécution et complexité asymptotique.
  - **Résultats des Tests** :

    json_vers_nx(chemin) : O(films*acteurs^2)

    collaborateurs_communs(G,u,v) : O(n)

    distance_acteurs(G,act1, act2) : O(n^3)

    centralite(G,u) : O(n^4)

    dist_naive(G,u,v) : O(k*n^2)

    est_proche(G,u,v,k=1) : O(k*n^2)

    distance_max_acteurs(G) : O(n^4)

    centre_groupe_acteurs(G,groupe) : O(groupe*n^4

- **Analyse des Résultats** :

Nous pouvons constater que le graphe est connexe et que les différents acteurs ont une proximité de maximum 7 films avec les autres acteurs. Ce qui nous permet de conclure que effectivement suite au différentes expérimentations faites pour l'analyse de ces données, la théorie du jeu des 6 degrés de Beacon.

.

5. Conclusion

Le projet a été réalisé avec succès en implémentant diverses fonctions de théorie des graphes pour analyser les collaborations entre acteurs. Les tâches ont été réparties de manière équilibrée entre les membres du binôme, ce qui a permis une collaboration efficace et une mise en œuvre réussie des fonctionnalités requises. Les réponses aux questions posées ont été obtenues en respectant les principes théoriques des graphes et ont été validées par une évaluation expérimentale rigoureuse.