Rapport de projet - longest trail

Rapport sur le problème du plus long chemin simple ou le problème de la plus longue chaîne.
Par Louis DECLERCK (TALE 5 - Numérique et Sciences de l'Informatique, M. Senot) @ https://github.com/Louis2675/

Version problème de décidabilité :
Problème : *Chemin de poids minorée*
Entrée : *G un graphe non orienté et pondéré et k un nombre entier*
Sortie : *existe-t-il un chemin de poids au moins k ne passant pas deux fois par un même sommet ?*

Version problème de calculabilité :
Problème : *Plus lourd chemin simple*
Entrée : *G un graphe non orienté et pondéré*
Sortie : *le plus lourd chemin ne passant pas deux fois par un même sommet*

Prérequis : Ce projet est basé sur un graphe pondéré non orienté. Il est donc nécessaire d'avoir des connaissances de base en théorie des graphes afin d'exploiter ce projet au maximum et les tests qui en ressortent.

Introduction

Le problème du plus long chemin simple contient deux versions différentes : **la calculabilité et la décidabilité**. Ces deux versions prennent place au sein de structures de données de type graphe.

- **La calculabilité** demande le plus long chemin ne passant pas deux fois par un même sommet.
- **La décidabilité** demande s'il existe un chemin de longueur au moins k ne passant pas deux fois par un même sommet

Ce projet utilise la **recherche exhaustive ou recherche par force brute** *(méthode algorithmique qui consiste principalement à essayer toutes les solutions possibles)*. Par exemple, pour trouver le maximum d'un certain ensemble de valeurs, on consulte toutes les valeurs.

Résolution du problème

La décidabilité demande s'il existe un chemin de poids au moins k ne passant pas deux fois par un même sommet.
La calculabilité demande quel est le chemin le plus lourd sans passer deux fois par un même sommet.

La résolution du problème de décidabilité se fait en plusieurs étapes :

1. On commence par prendre un graphe pondéré et non orienté.

Pour ce faire, on utilise la `classe Graphe` initialisée dans le fichier graphes.py. Dans cette classe, nous avons initialisé deux possibilités de graphe : un graphe donné par l'utilisateur en entrée pour l'objet et un graphe généré aléatoirement (c'est le paramètre random de la fonction __init__).

2. On génère tous les chemins possibles avec les sommets du Graphe en entrée.

Afin de générer tous les chemins, on utilise les fonctions permutations, permutations_de_taille_n et generer_parcours du fichier fonctions.py et on obtient donc alors une liste de tous les chemins possibles sur le graphe utilisé.

3. On teste l'ensemble des chemins pour tester leur validité

Une fois qu'on a l'ensemble des parcours possibles au sein du graphe, au sein d'un même parcours avec la fonction arrete_existe qui vérifie une arête puis on vérifie toutes les arêtes d'un même chemin avec test_arrete_parcours. Les chemins qui ne sont pas valides sont donc retirés et il ne nous reste plus que les chemins qui peuvent exister au sein du graphe

4. Calculer le poids maximal au sein du projet

Avec l'entièreté des parcours qui sont valides, on regarde la taille d'un parcours à l'aide de la fonction taille_parcours puis taille_maximale_parcours_valide, on calcule la taille maximale au sein des parcours valides du projet.

5. Décidabilité et Calculabilité

Après avoir obtenu la taille du chemin passant une seule fois par chaque sommet, il nous reste plus qu'à déterminer les cas de décidabilité et de calculabilité.

- Décidabilité : Au sein de la fonction decidabilite, on prend la taille maximale et on compare celle-ci avec k donné auparavant. Si la taille maximale au sein du graphe est supérieure à, le problème est donc vérifié, il existe donc bien un chemin de taille supérieure ou égale à k. Sinon le graphe ne vérifie pas le problème.

- Calculabilité : Dans la fonction calculabilité, on prend le graphe retourné par la fonction taille_maximale_parcours_valide, et on le retourne avec sa taille associée.

Conclusion

Applications de l'algorithme plus long chemin simple

- Chemin critique : La méthode du chemin critique pour l'ordonnancement d'un ensemble de tâches ou activités utilise la construction d'un graphe orienté acyclique : les sommets représentent des étapes du projet et les arcs des tâches qui doivent être accomplies entre une étape et la suivante ; chaque arc est pondéré par du temps nécessaire pour réaliser la tâche correspondante. Un plus long chemin de l'étape initiale à l'étape finale est un chemin critique, dont la longueur décrit le temps total nécessaire pour achever le projet. La méthode PERT *(program evaluation and review technique)* fait usage de cette notion, avec des développements considérables pour notamment inclure des éléments tenant compte des impondérables.

- Tracé de graphes : Le plus long chemin dans les graphes acycliques est utilisé dans le dessin de graphes par niveau : on attribue à chaque sommet v d'un graphe orienté acyclique G un niveau dont le numéro est la longueur du plus long chemin arrivant en v. On obtient ainsi une attribution de niveaux aux sommets de graphe avec un minimum de couches et qui a la propriété qu'un arc relie des sommets sur des couches de valeurs croissantes.

Ainsi, bien que cet algorithme ne fasse pas partie des recherches les plus importantes sur les graphes actuellement, il a ses usages dans le domaine de l'informatique.