
Nawras Mohammed Amin 1962832
Ragib Ahashan 1931613
Pritam Patel 1933097

# TRAVAIL PRATIQUE 1: GRAPHES

LOG2810 - STRUCTURES DISCRÈTES

**Remis le**
5 Novembre 2019

**Département de génie informatique et génie logiciel**
**Polytechnique Montréal**

## 1. Introduction

Le premier travail pratique du cours LOG2810 : Structure Discrètes était d'utiliser la théorie des graphes pour réussir à optimiser un problème donner.

Le problème que nous devions résoudre concernait la popularité grandissante de nos jours: les drones. Notre objectif était de trouver un moyen de rendre le parcours du drone aussi efficace que possible, compte tenu du nombre défini d'entrepôts situés sur le campus d'une entreprise et de trois objets différents que trois drones différents devraient transporter d'un entrepôt à l'autre. Essentiellement, nous avons dû implémenter l'algorithme « shortest path » de Dijkstra dans notre code afin d'extraire le chemin le plus court que le robot choisi devrait suivre pour aller du point A au point B.

Bien sûr, ce n'était pas si facile. Comme dans la vie réelle, de nombreux autres facteurs doivent être pris en compte. Des conditions telles que la masse des objets, le temps d'atterrissage et de montée des drones lors du ramassage d'objets, ainsi que le coût de certains chemins ont été indiqués et ont donc été considérées lors de l'implémentation de l'algorithme.

Ce rapport expliquera l'approche choisie pour résoudre le problème, ainsi qu'une explication des classes que nous avons conçues, grâce à un diagramme de classe. Finalement, nous irons à travers les points difficiles que nous avons rencontrés pendant la réalisation du projet.

## 2. Présentation des travaux

Pour commencer, nous avons décidé de faire le code complètement en python. C'était pour quelques raisons. D'abord, nous avons supposé que le fait de devoir allouer ou de désallouer de la mémoire manuellement n'était pas notre plus grande préoccupation. Cela éliminait les langages tels que C, Rust, etc. De plus, nous avons apprécié l'idée de disposer de librairies qui pourraient potentiellement nous donner une représentation graphique du graph plus « human-friendly », une fois venu le temps de l'imprimer. C'est-à-dire qu'au lieu de représenter le graph sur la ligne de commande, il existait des libraires python qui permettent d'utiliser du graphisme pour montrer le graph sur un Canvas.

En regardant le travail, nous avons décidé de procéder de manière orientée objet où nous aurions un objet Graph, Vertex, Arc, Commande, Drone, GestionnaireDeVols (FlightManager), ainsi que notre main. Nous expliquerons le comportement de chacune des classes.

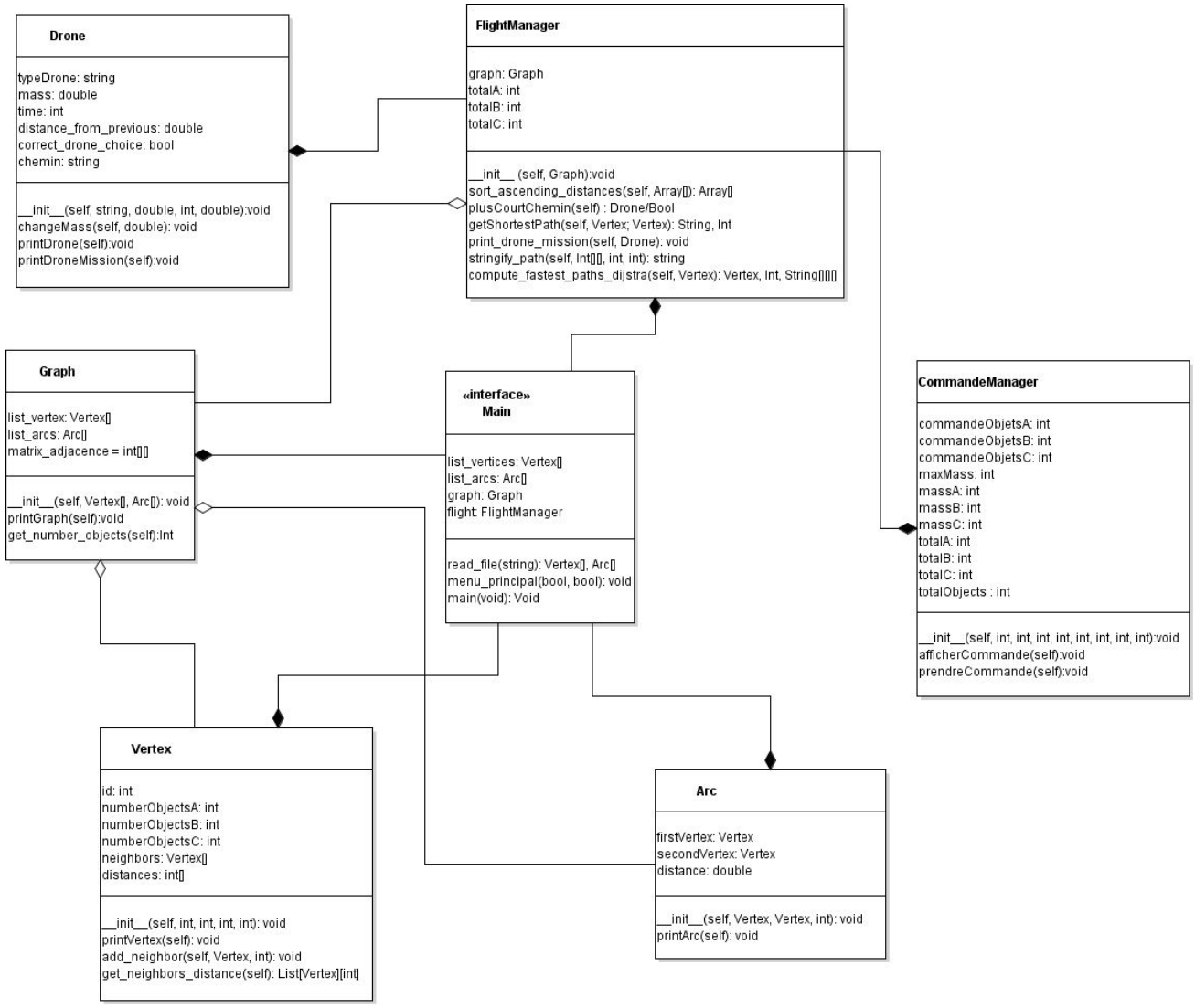


Figure 1 : Diagramme de Class du TP1.

Nous avons tout réalisé en gardant à l’esprit que cela devait fonctionner avec tout autre fichier représentant des entreprises. Dans cette mentalité, notre fonction read_file() permet de créer des listes de sommets et des listes d’arc de tous les fichiers ayant le même format que le fichier original. De plus, ces sommets et ces arcs permettent ensuite la création de n’importe quel graphe, en maintenant nos méthodes telles que printGraph uniformes pour tout type d’entreprise. Ensuite, Nous avons conçu la classe de Vertex dans laquelle nous pourrions stocker des propriétés spécifiques à utiliser ultérieurement, telles que le nombre d'objets, ainsi que les vertex voisins et les distances. La class Arc nous permet de savoir quel vertex sont liés ensemble. La classe

Graph est très important. Elle est utilisée dans FlightManager. La classe FlightManager prend le Graph et en extrait le nombre maximal d'objet A, B et C. Elle prend aussi un objet CommandeManager. La classe permet ensuite de générer le Drone qui serait le plus rapide pour la commande donne. Ce drone est ensuite utilisé dans la fonction print_drone_mission.

Le fichier Dijkstra.py contient la méthode compute_fastest_paths_dijstra. L'algorithme de la méthode a été inspiré par l'algorithme de Dijkstra pour le plus court chemin d'un sommet à un sommet. Après le calcul de la méthode, nous sommes en mesure de connaître la distance de chaque sommet du sommet au dernier sommet. Le stockage des valeurs de distance de chaque arc dans un tableau nous permet de les utiliser efficacement par la suite avec notre classe de gestionnaire de vol. Nous avons inclus les fonctions du fichier avec celles de FlightManager(Gestionnaire de Vol) dans le diagramme de classe, cependant nous avons décidé de les mettre dans un fichier séparé pour augmenter la cohésion.

Avant d'entrer dans le gestionnaire de vol, nous devons examiner Drone. La classe Drone contient trois méthodes, l'une consistant à imprimer, l'autre à modifier la masse totale que doit supporter le drone et enfin la dernière, celle du constructeur, qui initialise le type de drone en fonction de la masse indiquée dans paramètre. Les formules permettant de calculer les constantes K données dans le TP ont été prises en compte dans cette classe.

Enfin, la classe de gestionnaire de vol est très importante car elle gère les cas extrêmes qui nous ont été attribués, par exemple si la masse du robot dépasse 5,10 ou 25 ou les 10 secondes supplémentaires qui sont ajoutées lors de l'atterrissage. Comme mentionné, la classe prend un objet CommandeManager et un objet Graph. Les objets de commandeManager contiennent les objetsA, B et C que l'utilisateur veut que le drone aille chercher. La mission de vol prend ce commandement et renvoie le drone le plus rapide pour la mission. Il est capable de le faire en utilisant les équations mathématiques qui nous ont été données, ainsi qu'un objet de Dijkstra pour calculer le

temps le plus rapide à partir de la position de départ, jusqu'à ce que la commande soit complétée (ce qui signifie que le drone a ramassé tous les objets demandés). Le drone avec le temps le plus court est celui qui est retourné. Le drone est ensuite utilisé dans la méthode print_drone_mission où nous imprimons les informations pertinentes pour le vol et le drone.

Notre main contient l'interface où nous posons les choix de menu possible à l'usager. C'est-à-dire que l'usager peut choisir un fichier text à travers lequel le graph sera créer, L'usager peut ensuite imprimer le graph généré précédemment sur la ligne de commande. Il ou elle peut ensuite utiliser le menu pour donner la commande désirée et l'imprimer aussi par la suite. Finalement on peut aussi demander d'obtenir le chemin le plus proche, qui fait appel à nos fonctions expliquées ultérieurement.

## 3. Difficultés rencontrées :

Lors de la réalisation du projet, nous avons rencontré certaines difficultés. Pour commencer, notre décision de coder en python nous a mis sur la bonne voie au début. Cependant, l'un des membres de notre équipe était un débutant complet. Ce n'était pas idéal et malgré notre connaissance de cette nouvelle, nous avons décidé de continuer, sachant que cela pourrait être résolu.

Un autre problème auquel nous avons rapidement été confrontés était le manque de communication entre les membres. Cela nous a fait perdre de vue ce que nous faisons et notre objectif n'était plus aussi apparent. Heureusement, après s'être assis et en avoir discuté, tout allait bien.

En ce qui concerne un problème qui était davantage lié à la tâche à accomplir, après avoir réfléchi à la manière dont nous procédons, nous avons décidé de noter les cas à prendre en compte. Il y avait de nombreuses considérations. Nous nous sommes rendus compte qu'il nous faudrait procéder étape par étape avant de nous attaquer à cette énorme quantité d'informations.

## 4. Conclusion :

Dans l’ensemble, malgré des demandes assez difficiles, c’était un travail très intéressant et amusant. La théorie des graphes est un sujet très important en informatique. C’est une très bonne chose à savoir, en particulier pour les étudiants comme nous en deuxième année, qui essaient d’obtenir un stage, car cette connaissance est beaucoup recherchée dans les entreprises. Ce type de problème était non seulement très réaliste, ce qui le rendait d'autant plus intéressant mais aussi extrêmement satisfaisant à résoudre. Pour le prochain laboratoire, nous attendons un problème réaliste et stimulant pour tester la théorie que auront apprise en classe. Considérant que celui-ci était une surprise, nous voulons être prêts pour le prochain.