Rapport de Be Graphe

# Introduction

Nous avions comme objectif de comprendre et d’implémenter deux algorithmes de parcours de graphes ainsi que de réfléchir à un problème ouvert utilisant ses algorithmes de parcours. Les algorithmes de parcours sont l’algorithme de Dijkstra et l’algorithme A\*. Le problème ouvert est celui du covoiturage.

# Implémentation

Ce bureau d’étude ce veut le plus proche possible d’un projet que l’on pourrait avoir à développer en entreprise. Afin de pouvoir coder sur le même projet en même temps nous avons utilisé un logiciel de gestion de version, git.

La structure du programme était déjà donnée, il ne nous restait qu’à implémenter les algorithmes et les quelques classes dont ils avaient besoin.

Nous avons implémenté un tas pour ensuite pouvoir développer l’algorithme de Dijkstra.

Nous avons pu découvrir le concept de design pattern Observer qui permet de notifier des objets d’autres classes sans devoir multi-threader notre programme.

Lors des tests sur la carte « carré dense » nous pouvons voir le comportement de l’algorithme de Dijkstra qui se répand dans toutes les directions. A contrario, l’algorithme A\* suit une ligne directrice dans la direction de la destination.

# Tests de corrections

Nous avons pu prendre en main le Framework JUnit qui permet de tester notre programme de manière automatique et non pas de seulement de faire des affichages de nos résultats. Nous avons réalisé tous nos tests grâce à ce Framework.

Nous avons mené divers tests de corrections portant sur des cartes fictives et réelles. Ces tests sont validés. Les tests portaient sur des chemins les plus courts en distance ainsi que sur la recherche de chemin les plus rapides en temps. Nous les avons faits pour l’algorithme de Dijkstra et A\*.

# Tests de performances

Nous avons repris les tests de corrections auxquels nous avons ajouté deux appels system pour connaitre le temps d’exécution des tests pour chaque algorithme. Nous avons pu remarquer que l’algorithme de Bellman-Ford est le plus lent suivi de l’algorithme de Dijkstra, enfin l’algorithme A\* est le plus rapide.

# Problème ouvert

Nous avons choisi le problème du covoiturage. Deux usagers veulent covoiturer, nous devons trouver le point de rencontre optimal pour qu’ils effectuent le moins de kilomètres. Ils peuvent faire le trajet chacun de leur côté dans le pire des cas.

On a représenté ce problème sous la forme d’un triangle, avec pour sommets les usagers et la destination. Nous avons trouvé que le point de rencontre optimal s’il existe est le centre de gravité de ce triangle.

Nous trouvons donc le chemin le plus rapide entre les deux usagers. S’il est plus long que le chemin de la somme des trajets entre les usagers et la destination alors nous décidons que le covoiturage n’est pas une solution optimale.

Ensuite nous calculons le point qui est le plus proche de la moitié du coût du chemin le plus court entre les deux usagers, appelé Intersection.

Nous trouvons le plus court chemin entre intersection et la destination et le centre de gravité ce trouvera à 1/3 de distance à partir de intersection.

Le chemin optimal pour chaque usager est donc le trajet jusqu’au centre de gravité puis jusqu’à leur destination.