EL HAJ HUSSEIN Thomas
LIEFFROY Théo

# RAPPORT PROJET AG44

---

## *Introduction*

Lors de l'UV AG44, Graph Theory and Applications, il nous a été demandé d'effectuer un projet informatique permettant de mettre en pratique les cours sur la théorie des graphes par le biais d'algorithmes.

Ce projet a été réalisé en C++qui est un langage de programmation orienté objet permettant de définir des classes et d'y attribuer des méthodes.

## *Structure du programme*

Comme dit précédemment, la puissance du C++ réside dans le fait de pouvoir définir des classes possédant des attributs et qui interagissent entre elles à travers des méthodes.

Dans une première partie nous avons défini les quatre classes principales du projet :

- Vertex : une classe représentant les sommets d'un graphe contenant son ID pour le différencier des autres ainsi qu'une liste de ses voisins

- Edge : une classe représentant les arêtes d'un graphe contenant son ID , son potentiel coût ainsi que le sommet de départ et celui d'arrivé

- Graph : une classe représentant le graphe en lui-même contenant alors une liste de sommets et les arêtes. Il contient également son ID

Vient alors la deuxième partie c'est-à-dire l'implémentation des différentes méthodes nécessaires à la génération des graphes.

Tout d'abord nous avons implémenté la génération aléatoire et automatique d'un graphe à n vertex grâce à un constructeur de la classe Graph et des constructeurs de Edge et Vertex.

Nous avons aussi implémenté une méthode à Graph permettant de récupérer un graphe pré-construit sur un fichier externe au programme et également d'écrire le graphe sur un fichier externe formaté.

Il est possible d'afficher le graphe sur la console en matrice et en liste.

La troisième partie consistait à implémenter les algorithmes vus en cours pour leurs applications.

Nous avons commencé par intégrer l'algorithme Breadth First Search raccourci en BFS.

Il s'agit d'un algorithme de recherche dans un graphe qui balaie en largeur dans un graphe tous les sommets liés à un sommet de départ.

Ce fût ensuite le tour de l'algorithme de Depth First Search raccourci en DFS, il s'agit ici d'un algorithme de recherche dans un graphe privilégiant la profondeur à la largeur à partir d'un sommet de départ, ce qui ne l'empêche pas de parcourir tous les sommets liés au sommet de départ.

Le DFS étant implémenté, nous nous en sommes servi pour pouvoir implémenter l'algorithme suivant : l'algorithme de recherche des composantes fortement connexes.

Cet algorithme utilise une structure de type « stack » ainsi que le DFS pour lister les groupes de sommets fortement connectés entre eux.

Ensuite nous avons implémenté le tri topologique grâce encore au DFS.

## *Problèmes rencontrés*

Le premier problème que nous avons rencontré lors de l'élaboration d'un projet en C++, qui était une première pour nous. En effet nous n'avions jamais travaillé en C++ auparavant. Même si nous avions quelques notions de langage de programmation orienté objet, au début il a été assez difficile de travailler avec un langage sur lequel nous n'étions pas à l'aise d'autant plus que le langage C++ a quelques subtilités qui diffère bien des autres langages, comme par exemple l'allocation de mémoire en C++ est différente de celle en Java. Nous avons donc opté pour l'utilisation de pointeurs au milieu du projet, ce qui nous a également ralentis.

Le choix des méthodes d'implémentation, par exemple une « queue » pour le BFS et un « stack » pour le DFS, n'était pas simple à comprendre directement au début et il a fallu un peu se creuser les méninges pour que l'adaptation informatique fonctionne.

La gestion du temps pour terminer le projet a également été un problème, quelques fois un petit problème de dépassement de mémoire à pu nous coûter plusieurs heures pour comprendre d'où venait l'erreur.

## *Conclusion*

Ce projet nous a permis de mieux comprendre les applications des notions relatives à la théorie des graphes ainsi qu'à utiliser le langage C++. Il nous a également permis de nous familiariser avec GitHub pour le travail en groupe.

Lors des recherches pour implémenter les différents algorithmes, nous avons pu avoir une nouvelle approche sur les graphes et trouver des applications intéressantes de résolutions de problèmes, comme par exemple trouver un chemin dans un labyrinthe avec le DFS ou encore le chemin le plus court avec l'algorithme de Dijkstra.