SAE S2.02 -- Rapport pour la ressource Graphes

Auteurs : Yanis.E, Vincent.A, Yannis.D | Groupe F.

Version 1 : un seul moyen de transport

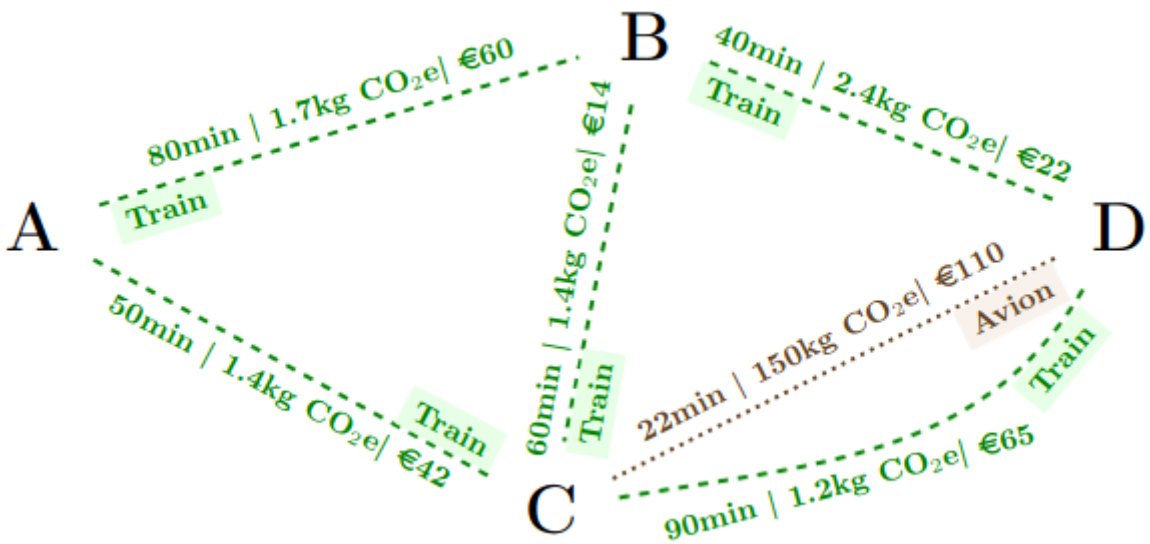
Présentation d'un exemple

En utilisant le graphe présenté si dessous, imaginons une utilisatrice nommée Claire qui doit se rendre à une conférence à D. Elle habite à A et souhaite trouver l'un des meilleur itinéraire possible. Les moyens de transport disponibles incluent le train et l'avion. Elle souhaite trouver plusieurs itinéraire répondant le plus possible à ses préférences.

Les préférences de l'utilisateur doivent pouvoir être choisis en fonction de ma modalité de transport souhaiter, du nombre de destination et d'un ou plusieurs critères de voyage tel que le temps du trajet, le prix ou encore le rejet en CO2.

Par exemple, si elle choisis de ne prendre que le train pour ne pas polluer, et avoir les trajets les moins couteux, le programme ne calculera que des trajet par train les moins chère possible de A vers D. On peut ajouter un critère secondaire tel que le temps ou le CO2.

Modèle pour l'exemple



Ici, pour aller de A vers D comme Claire en souhaitant le chemin le plus court, c'est à dire le plus rapide, ce serait le chemin [A,B,D] pour lequel le temps est de 120 minutes.

Modélisation pour la Version 1 dans le cas général

Pour construire un graphe permettant de résoudre le problème de recherche d'itinéraire, il nous faut :

- plusieurs sommets qui sont représentés par des villes en l'occurence : A,B,C,D.
- Plusieurs arêtes qui permettent de relier les villes entre elles, ici :

    - A - B (Train)
    - A - C (Train)
    - B - C (Train)
    - B - D (Train)
    - C - D (Avion)
    - C - D (Train)

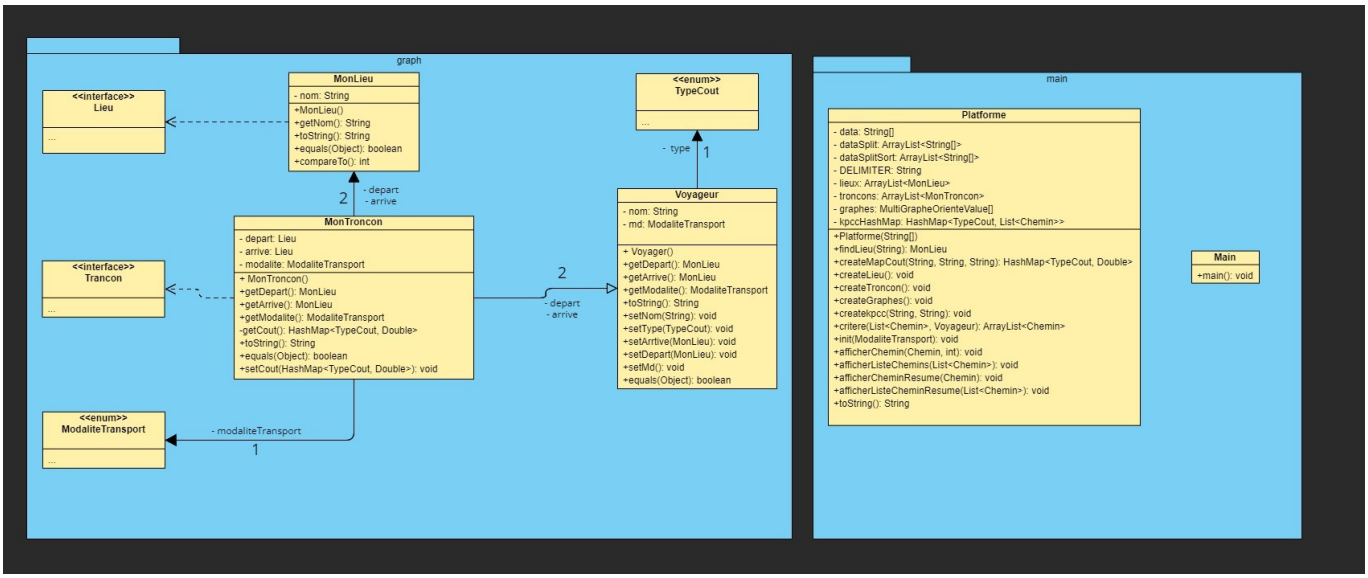
- Le poid des arêtes sont définis par 3 critères au choix qui sont :

    - Le temps exprimé en minutes.
    - Le rejet de CO2 exprimé en kg CO2e.
    - Le prix du trajet en euros. (€)

- L'algorithme utilisé pour le calcule du plus court chemin (pcc) est l'algorithme de Dijkstra qui prend un graphe orienté ou non et valué positif en entrée.

Implémentation de la Version 1

Pour modéliser notre projet, nous avons réaliser un diagramme UML représentant chaque classe à implémenter.



ClaireTest.java

Une de nos classes de test s'appelle **PlatformeTest.java** et permet construire un graphe et de tester certaine méthodes de Platforme.java.

Il y a aussi **ClaireTest.java** qui lui construit un graphe repris de l'exemple de Claire pour ensuite pouvoir calculer et renvoyer le bon chemins.

- Date du commit : 20/05/2024
- Lien du commit : bbd43acf5f1d796be56b6849a635ef0645146289
- Lien du fichier sur Gitlab

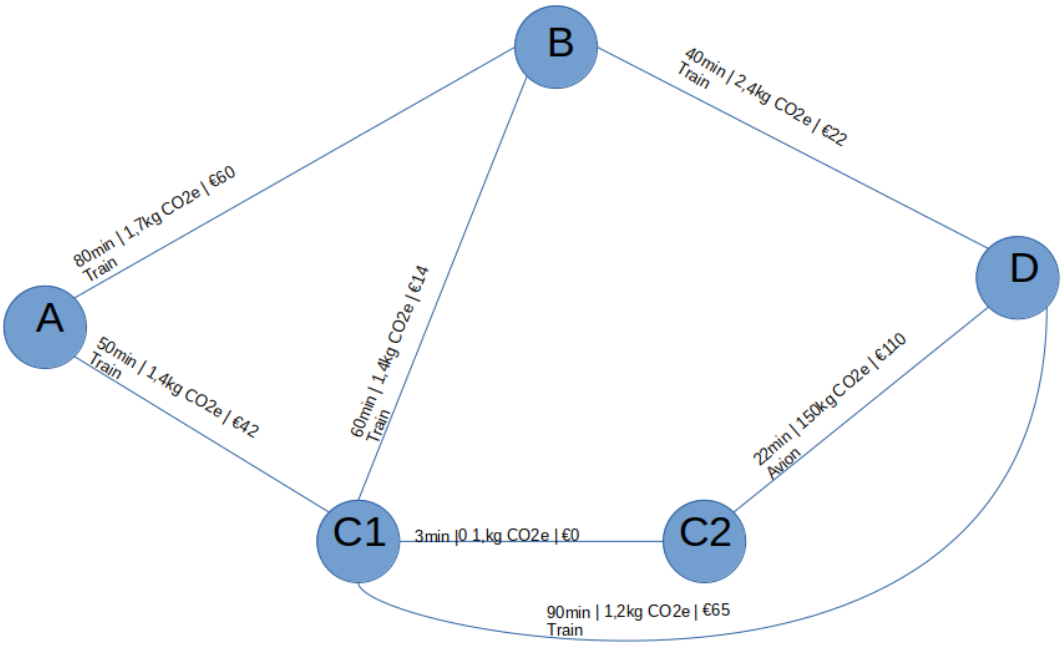
Version 2 : multimodalité et prise en compte des correspondances

Dans cette seconde version, il fallait rajouter des correspondances dans les villes. De ce fait, nous avons rajouté une **List de tableau de String** afin de pouvoir stocker et manipuler chaque correspondance. Nous l'avons ajouté dans la classe **MonLieu**, car chaque correspondance se trouve dans la ville.

Présentation d'un exemple

Finalement, la conférence de Claire a été avancée. Elle décide donc de se rendre à la ville D le plus rapidement possible. De ce fait, elle décide de prendre l'avion de manière exceptionnelle. La ville C possède une gare et un aéroport. Claire a rajouté la modalité de transport **"Avion"** dans son itinéraire donc une correspondance a été réalisé entre la gare et l'aéroport.

Modèle pour l'exemple



- Le sommet C a été dupliqué :
  - C1 et C2
- Ce qui crée une nouvelle arrête :
  - C1 - C2 (Correspondance)

L'itinéraire optimal proposé à Claire est de partir de la ville A passant par la ville C, marcher jusqu'à l'aéroport de la ville C et prendre l'avion pour arriver à la ville D. Pour avoir le chemin le plus optimal, Claire décide donc de marcher de la gare à l'aéroport de la ville C. D'autre itinéraires sont proposés, mais le plus optimale reste celui où il faut utiliser l'avion.

Modélisation pour la Version 2 dans le cas général

L'utilisateur pourra rentrer deux critères de sélection afin de pouvoir visualiser quel graphe est le plus adapté au critère demandé. Ce critère s'applique sur tous les chemins y compris les correspondances.

Pour calculer quel chemin est le plus adapté aux critères demandé, nous avons additionné le coût des critères sélectionne en y comptant aussi les correspondances.

Implémentation de la Version 2

- Nom de la classe : **ClaireTest2.java**
- Date : **7/06/2024**
- ID Commit : **11678edf7a3b999cb5690047476ac6b5d9c3183d**
- URL Commit : **https://gitlab.univ-lille.fr/sae2.01-2.02/2024/F4/-/blob/11678edf7a3b999cb5690047476ac6b5d9c3183d/test/main/ClaireTest2.java**

Version 3 : optimisation multi-critères

Fin du rapport

Barème sur 30 pts

Toute question sur le barème est à adresser à iovka.boneva@univ-lille.fr

- Rapport non rendu à temps -> note 0
- **(7, décomposé comme suit)** Divers
  - **(1,5)** Respect de la structure du rapport
  - **(1,5)** Section Version 1 rendue pour le 18/05/2024. Cette version peut contenir les parties en italique.
  - **(1,5)** Section Version 2 rendue pour le 08/06/2024. Cette version peut contenir les parties en italique.
  - **(1)** Utilisation de vocabulaire précis sur les graphes (termes vu en cours, noms des algorithmes, etc.)
  - **(1,5)** Style d'écriture fluide et compréhensible
- **(8, décomposé comme suit)** Solution pour la Version 1
  - **(2)** Exemple pertinent (illustre tous les aspects du problème) et lisible (en particulier, ni trop grand ni trop petit, bien présenté)
  - **(4)** Le modèle de l'exemple permet de trouver la solution sur l'exemple. La modélisation pour le cas général permet de résoudre le problème posé
  - **(2)** L'implémentation de l'exemple est correcte et fonctionnelle
- **(6, décomposé comme suit)** Solution pour la Version 2
  - **(1)** Exemple pertinent
  - **(4)** le modèle de l'exemple permet de trouver la solution sur l'exemple. La modélisation pour le cas général permet de résoudre le problème posé
  - **(1)** L'implémentation de l'exemple est correcte et fonctionnelle
- **(3)** Qualité de la description de la solution (concerne les sections "Modèlisation dans le cas général" pour les Versions 1 et 2):
  - La modélisation pour le cas général est décrite de manière abstraite mais précise et complète. Pour vous donner une idée, un·e étudiant·e de BUT qui a validé les ressources Graphes et Dev devrait être en mesure d'implémenter votre solution d'après la description que vous en faites, sans avoir à trop réfléchir.
- **(6)** Solution pour la Version 3: mêmes critères que pour la Version 2