UNIVERSITÉ LIBRE DE BRUXELLES

INFO-F203 - Projets d'algorithmique 2

Projet Mobilité

Auteurs :
Romain Liefferinckx - 000591790
Manuel Rocca - 000596086

Professeurs :
Jean Cardinal
Assistants :
Robin Petit

Année académique 2024-2025

Table des matières

1 Introduction

Dans le cadre de notre cours d'algorithmique INFO F-203, l'occasion s'est présentée à nous de créer un programme cherchant un chemin optimal entre un point A et un point B sur base d'une heure de départ. En effet, sur base d'un ensemble de données fournies sous format *General Transit Feed Specification* (**GTFS**), nous avons utilisé le *Connexion Scan Algorithm* (**CSA**) pour implémenter notre chercheur de chemin en *Java*.

Dans les sections à suivre, nous abordons le parsing des données, des structures formées à partir de celles-ci pour notre implémentation et certains détails techniques comme la complexité temporelle et spatiale. Nous justifierons également certains choix comme celui de l'algorithme précisé ci-dessus, à savoir le **CSA**.

2 Parsing

Dans cette section nous expliquons les procédés utilisés pour charger les données en mémoire à partir des fichiers CSV fournis ainsi que les structures de données utilisées pour leur stockage et leur utilisation optimale dans l'algorithme choisi par nos soins.

Nous utilisons une classe Parser, qui se charge de la lecture des fichiers CSV. Cette classe est responsable de la création des objets de type *Connexion*, *Route*, *Stop* et *Trip* à partir des fichiers. Qui nous permettent de créer les connexions entre arrêts, les routes et les trajets avec les données des fichiers csv.

2.1 Structures de données pour le stockage

Dû à notre choix d'implémentation algorithmique, nous avons opté pour des structures efficaces pour utiliser l'utiliser dans les meilleurs conditions possibles. En effet, le CSA, comme son nom le suggère, fait une forte utilisation des connexions entre arrêts, chose que nous détaillons plus loin dans ce rapport.

D'abord ont été créées les quatre classes principales, chacune correspondant à un type de fichier CSV donné. De manière générale, chaque champ de chaque fichier est repris comme un attribut de classe. Cela n'est cependant pas toujours le cas, nous le précisons dans les sections adéquates.

2.1.1 La classe Route

Cette classe est une simple classe de stockage, chaque attribut correspondant à un champ des fichiers *routes.csv*.

Attribut	Type	Description
routeId	final String	L'id de la route représentée
routeShortName	final String	Le nom de la route raccourci
routeLongName	final String	Le nom de la route complet
routeType	final String	Le type de véhicule utilisant cette route

Table 1 – Classe Route

2.1.2 La classe StopTime

La particularité de cette classe est qu'il lui manque le champ tripId donné dans les CSV concernés. Ce choix découle de la structure de la classe Trip détaillée dans la section 2.1.4.

Attribut	Type	Description
departureTime	final String	L'heure du départ à partir de l'arrêt associé sur le trajet associé en format heure ;minutes ;secondes
stopId	final String	L'id de l'arrêt associé
stopSequence	final int	Le numéro de l'arrêt dans le trajet associé

Table 2 – Classe StopTime

2.1.3 La classe Stop

Dans cette classe, hormis le fait que chaque champ des fichiers *routes.csv* est repris, nous avons fait le choix d'ajouter une liste de tripId, permettant de retrouver efficacement chaque Trip partant de cet arrêt.

Attribut	Type	Description
stopId	final String	L'id du stop représenté
stopName	final String	Le nom du stop
stopLat	final String	La latitude du stop
stopLon	final String	La longitude du stop
trip_ids	List<String>	La liste des tous les trips (leurs ids) partant de ce stop

Table 3 – Classe Stop

2.1.4 La classe Trip

Une fois de plus, en plus des champs trouvés dans les fichiers *trips.csv* retranscrits en attribut, nous avons ajouté une liste de StopTime associés à ce Trip. Ceci nous permet de construire efficacement les connexions entre arrêts affublées de temps de départ et d'arrivée.

Attribut	Type	Description
tripId	final String	L'id du trip représenté
routeId	final String	L'id de la route sur laquelle il passe
stopTimes	List<StopTime>	La liste de tous les StopTime associés à ce trip

Table 4 – Classe Trip

3 Le Connexion Scan Algorithm

4 Sources-bibliographie