Projet Structures de données : avancé

1 Objectif général du projet

Le fichier `stib.xml` contient des données à propos du réseau de la STIB (Société des Transports Intercommunaux de Bruxelles).

Ces données forment un graphe dirigé. Les sommets sont les stations. Il y a un arc entre deux stations s'il existe un tronçon reliant un des arrêts (stop) des deux stations.

L'objectif général du projet est d'implémenter un programme java permettant de calculer des itinéraires entre deux stations. Votre programmera sauvegardera les itinéraires calculés dans un fichier au format xml. Voici un exemple d'itinéraire :

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<trajet depart="MALIBRAN" destination="ALMA" duree="40"
        nbTroncons="14">
        <deplacement arrivee="SCHUMAN" attenteMoyenne="6"
                depart="MALIBRAN" direction="AMBIORIX" duree="13" nbTroncon="6"
                type="bus">60</deplacement>
        <deplacement arrivee="ALMA" attenteMoyenne="8"
                depart="SCHUMAN" direction="STOCKEL" duree="13" nbTroncon="8"
                type="metro">1</deplacement>
</trajet>
```

Ceci représente un trajet de la station MALIBRAN à la station ALMA. La durée du trajet est de 40 minutes (temps de déplacement + temps d'attente moyen). Il est constitué de deux déplacements : un déplacement avec le bus 60 (direction AMBIORIX) de la station MALIBRAN à la station SCHUMAN et un deuxième avec le métro 1 (direction STOCKEL) de la station SCHUMAN à la station ALMA.

Remarquez qu'il ne faut pas préciser les stations intermédiaires si on ne change pas de ligne. De plus, l'attribut `arrivee` d'un déplacement sera toujours le même que l'attribut `depart` du déplacement suivant (s'il existe). Les itinéraires que vous calculerez devront respecter ce format.

2 Sur Moodle

Pour mener à bien votre projet, nous vous fournissons plusieurs fichiers :

- `stib.xml` contenant les informations sur le réseau de la STIB.
- `stib.xsd`, un schéma xml permettant de valider `stib.xml`.
- une classe `Main.java` que vous ne pouvez pas modifier. Le code de cette classe est présenté ci-dessous. La classe à construire `Graph` devra contenir deux méthodes pour calculer des itinéraires. Ces deux méthodes seront détaillées dans la suite du document. Ces méthodes prennent trois paramètres. Les deux premiers paramètres sont les noms des stations de départ et d'arrivée. Le troisième paramètre est le nom du fichier où il faudra sauvegarder le fichier xml de résultat.

```java
import java.io.File;
import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory;

public class Main {
  public static void main(String[] args) {
    try {
      File inputFile = new File("stib.xml");
      //File inputFile = new File("stibattente0.xml");
      SAXParserFactory factory = SAXParserFactory.newInstance();
      SAXParser saxParser = factory.newSAXParser();
      SAXHandler userhandler = new SAXHandler();
      saxParser.parse(inputFile, userhandler);
      Graph g = userhandler.getGraph();
      g.calculerCheminMinimisantNombreTroncons("MALIBRAN", "ALMA", "output.xml");
      g.calculerCheminMinimisantTempsTransport("MALIBRAN", "ALMA", "output2.xml");}
    catch (Exception e) {e.printStackTrace();}
  }
}
```

3 Tâches à effectuer

3.1 Tâches obligatoires

Nous vous demandons de rendre la classe `Main` fonctionnelle en réalisant les tâches suivantes :

- Implémenter un parseur SAX permettant la lecture du fichier XML et la construction du graphe en mémoire en utilisant les structures de données adéquates.
- Implémenter la méthode `calculerCheminMinimisantNombreTroncons` qui calcule le chemin entre deux stations avec le moins de tronçons possibles (voir point 4). S'il est impossible d'aller d'une station à une autre, votre programme lancera une exception.
- Implémenter la méthode `calculerCheminMinimisantTempsTransport` qui calcule le chemin entre deux stations avec le moins de temps passé dans les transports (voir point 5). S'il est impossible d'aller d'une station à une autre, votre programme lancera une exception.
- Sauvegarder ces chemins dans des fichiers XML.

De plus, nous vous demandons d'écrire la DTD `stib.dtd` qui permet de valider les mêmes documents que `stib.xsd` (tout en étant plus permissive évidemment).

3.2 Tâches bonus

Si vous avez fini trop tôt et que vous vous embêtez durant les séances, vous pouvez faire une ou plusieurs tâches bonus. L'objectif est de faire le projet durant les séances, ne prenez pas du temps à la maison pour faire ces tâches bonus ; privilégiez les autres cours comme le projet Projet AE.

Voici les différentes tâches bonus :

1. Implémentez un parseur DOM pour créer le graphe.
2. Donnez la DTD permettant de valider les chemins calculés. Pour ajouter la dtd dans un fichier xml, voici une petite aide : https://stackoverflow.com/questions/13553614/how-to-add-doctype-in-xml-document-using-dom-java
3. Ajoutez une troisième façon de calculer un chemin en minimisant le temps total (voir point 6)

4 Itinéraire avec le moins de tronçons

La méthode `calculerCheminMinimisantNombreTroncons` calculera les itinéraires qui minimise le nombre de tronçons.

Par exemple, pour aller de la station MALIBRAN à la station ALMA, l'itinéraire le plus court contient 12 troncons.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<trajet depart="MALIBRAN" destination="ALMA" duree="127"
        nbTroncons="12">
        <deplacement arrivee="BLYCKAERTS" attenteMoyenne="6"
                depart="MALIBRAN" direction="AMBIORIX" duree="2" nbTroncon="1"
                type="bus">60</deplacement>
        <deplacement arrivee="IDALIE" attenteMoyenne="36"
                depart="BLYCKAERTS" direction="GARE CENTRALE" duree="2" nbTroncon="1"
                type="bus">N09</deplacement>
        <deplacement arrivee="LUXEMBOURG" attenteMoyenne="5"
                depart="IDALIE" direction="GARE CENTRALE" duree="2" nbTroncon="1"
                type="bus">38</deplacement>
        <deplacement arrivee="SCHUMAN" attenteMoyenne="6"
                depart="LUXEMBOURG" direction="BRUSSELS AIRPORT" duree="6"
                nbTroncon="1" type="bus">12</deplacement>
        <deplacement arrivee="MERODE" attenteMoyenne="8"
                depart="SCHUMAN" direction="STOCKEL" duree="2" nbTroncon="1"
                type="metro">1</deplacement>
        <deplacement arrivee="MONTGOMERY" attenteMoyenne="33"
                depart="MERODE" direction="MUSEE DU TRAM" duree="2" nbTroncon="1"
                type="bus">N06</deplacement>
        <deplacement arrivee="ALMA" attenteMoyenne="8"
                depart="MONTGOMERY" direction="STOCKEL" duree="9" nbTroncon="6"
                type="metro">1</deplacement>
</trajet>
```

Il est peut-être possible de trouver d'autres itinéraires avec 12 troncons.

5 Itinéraire avec le moins de temps passé dans les transports

La méthode `calculerCheminMinimisantTempsTransport` calculera les itinéraires qui minimise le temps passé dans les transports. Ce calcul ne tiendra pas compte des attentes moyennes. Vous pouvez utiliser le fichier stibattente0.xml pour tester votre programme.

Sur le fichier stibattente0.xml, pour aller de MALIBRAN à ALMA, il est possible de trouver un itinéraire avec 20 minutes passés dans les transports.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<trajet depart="MALIBRAN" destination="ALMA" duree="20"
        nbTroncons="13">
        <deplacement arrivee="BLYCKAERTS" attenteMoyenne="0"
                depart="MALIBRAN" direction="GARE CENTRALE" duree="1" nbTroncon="1"
                type="bus">N09</deplacement>
        <deplacement arrivee="VARIA" attenteMoyenne="0"
                depart="BLYCKAERTS" direction="AMBIORIX" duree="2" nbTroncon="1"
                type="bus">60</deplacement>
        <deplacement arrivee="JOURDAN" attenteMoyenne="0"
                depart="VARIA" direction="AMBIORIX" duree="2" nbTroncon="1" type="bus">60
        </deplacement>
        <deplacement arrivee="SCHUMAN" attenteMoyenne="0"
                depart="JOURDAN" direction="MUSEE DU TRAM" duree="2" nbTroncon="2"
                type="bus">N06</deplacement>
        <deplacement arrivee="MERODE" attenteMoyenne="0"
                depart="SCHUMAN" direction="STOCKEL" duree="2" nbTroncon="1"
                type="metro">1</deplacement>
        <deplacement arrivee="MONTGOMERY" attenteMoyenne="0"
```

```
                depart="MERODE" direction="MUSEE DU TRAM" duree="2" nbTroncon="1"
                type="bus">N06</deplacement>
        <deplacement arrivee="ALMA" attenteMoyenne="0"
                depart="MONTGOMERY" direction="STOCKEL" duree="9" nbTroncon="6"
                type="metro">1</deplacement>
</trajet>
```

6 Itinéraire minimisant le temps (Bonus)

Si vous avez le temps, vous pouvez calculer les itinéraires minimisant le temps total.

Avec les stations comme sommets du graphe, il est impossible d'utiliser l'algorithme de Dijkstra car les coûts ne sont pas seulement sur les arcs mais également sur les sommets.

Si on veut utiliser l'algorithme de Dijkstra pour calculer l'itinéraire minimisant le temps, il faudra donc changer la structure de graphe. Créez donc un nouveau projet pour ne pas polluer les méthodes qui fonctionnent ; si vous avez fait ce bonus, vous soumettrez deux projets différents sur moodle.

Pour chaque station, le nouveau graphe contiendra autant de sommets que le nombre de tronçons entrants dans cette station.

Cette tâche est légèrement plus compliquée que les autres. Si vous voulez plus d'indications, n'hésitez pas à poser vos questions aux professeurs.

Voici l'itinéraire minimisant le temps entre MALIBRAN et ALMA :

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<trajet depart="MALIBRAN" destination="ALMA" duree="40"
        nbTroncons="14">
        <deplacement arrivee="SCHUMAN" attenteMoyenne="6"
                depart="MALIBRAN" direction="AMBIORIX" duree="13" nbTroncon="6"
                type="bus">60</deplacement>
        <deplacement arrivee="ALMA" attenteMoyenne="8"
                depart="SCHUMAN" direction="STOCKEL" duree="13" nbTroncon="8"
                type="metro">1</deplacement>
</trajet>
```

7 Organisation et livrables

Ce projet se déroule du 4 mars 2018 au 22 mars 2018 par groupe de deux étudiants. Vous composerez vous-même les groupes en respectant la règle que tous les étudiants d'un même groupe doivent être dans la même série. Vous nous communiquerez la composition des groupes au début de la semaine du 3 mars. Durant tout le projet, la présence aux cours est obligatoire.

Si une série comporte un nombre impair d'étudiant, nous accepterons un seul groupe de 3 étudiants. Ce groupe de 3 étudiants devra également faire les deux premières tâches bonus.

Le projet est à remettre via « Moodle » pour le dimanche 22 mars 2018 à 12h00 (midi). Nous ne demandons pas de rapport. Si certaines choses méritent des explications, vous pouvez les fournir en commentaire dans les différents fichiers.

Le plagiat sera lourdement sanctionné (au minimum 0 à l'UE). Tous les projets seront automatiquement testés par un logiciel de détection de plagiat.