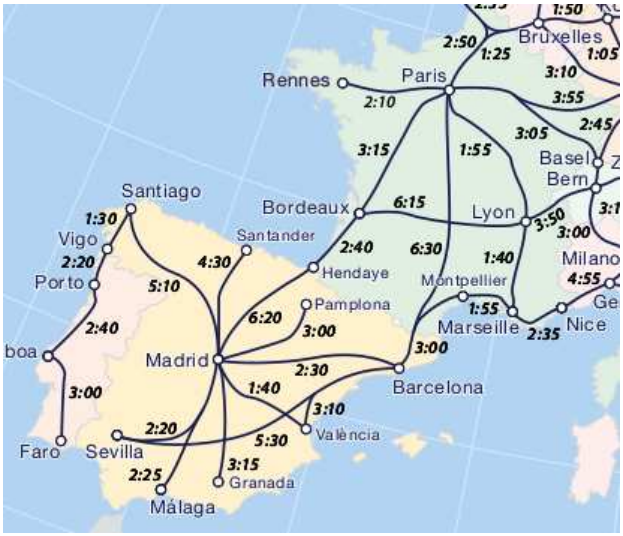
TD 21 : Algorithme de Dijkstra en OCaml

Réseau ferroviaire

On s'intéresse à une partie du réseau ferroviaire européen, sur lequel on va mettre en oeuvre l'algorithme de Dijkstra afin de trouver les trajets le plus court reliant une ville aux autres.



Dans le fichier `td_disjkstra.ml`, on définit les types OCaml suivants :

```
type sommet = int

type distance = int

type graphe = (sommet * distance) list array
```

Lecture à partir de fichier en OCaml

En pratique, les données volumineuses sont souvent stockées dans des fichiers de format donné, qui sont ensuite lus au démarrage du programme pour être manipulées informatiquement.

Ici le fichier `villes.txt` contient en première ligne le nombre de villes du réseau, et ensuite le nom de chacune des villes du réseau.

Le fichier `liaisons.txt` contient la description du réseau : le nombre de villes, le nombre d'arêtes, puis la description de chaque arête pondérée par la distance du trajet en minutes.

```
9
Seville
Madrid
Hendaye
Bordeaux
Paris
Lyon
Marseille
Montpellier
Barcelone
```

```
9
12
0 1 140
0 8 330
1 2 380
2 3 160
3 4 195
4 5 115
3 5 375
...
```

Q1. En utilisant les fonctions du "Mémo OCaml", écrire une fonction `lire_villes:string->string array` telle que `lire_villes nom_de_fichier` renvoie le tableau des noms de villes contenu dans le fichier `nom_de_fichier` (au format identique à `villes.txt`)

Q2. En utilisant la fonction précédente, définir une variable globale `villes:string array` contenant le tableau des villes de `villes.txt`

Q3. Ecrire une fonction `lire_reseau:string->graphe` telle que `lire_reseau nom_de_fichier` renvoie le graphe correspondant au contenu du fichier `nom_de_fichier` (au format identique à `liaisons.txt`)

NB : la fonction définie `lireArete:string->int*int*int` permet d'obtenir les trois valeurs à partir d'une ligne décrivant une arête pondérée.

Q4. En utilisant la fonction précédente, définir une variable globale `reseau:graphe` correspondant au graphe décrit par `liaisions.txt`

Q5. Ecrire une fonction `afficheGraphe:unit->unit` qui utilise les variables globales `villes` et `reseau` pour affiche les informations concernant le réseau ferroviaire.

(★ : **appel professeur**)

```
Seville
--> Barcelone (330 min)
--> Madrid (140 min)
Madrid
--> Barcelone (150 min)
--> Hendaye (380 min)
--> ...
```

File de priorité en OCaml

L'algorithme de Dijkstra utilise idéalement une `file de priorité`, pour laquelle il n'existe pas de module prédéfini en OCaml

Une implémentation (assez complexe, mais compréhensible) est ici fournie. On implémente une structure de données `(int*int) tas` permettant de stocker des couples (distance, sommet) et d'en extraire efficacement le couple dont la distance est la plus petite.

On sera amené à utiliser les fonctions suivantes :

- `creer_tas:unit->(int*int) tas`
- `ajouter:(int*int) tas->int*int->unit`
- `est_vide:(int*int) tas->bool`
- `retirer_min:(int*int) tas->int*int`

Q6. Tester cette implémentation : créer un tas vide, y ajouter 10 couples de valeurs entières aléatoires entre 0 et 100. Retirer et afficher tous les éléments du tas. Dans quel ordre doivent-ils apparaître ? (★ : **appel professeur**)

Algorithme de Dijkstra

Q7. Ecrire la fonction `dijkstra:graphe->sommet->int array` telle que `dijkstra graphe s` renvoie le tableau des distances minimales du sommet `s` à tout autre sommet. Afficher les distances obtenues pour le sommet 0 (★ : **appel professeur**)
NB : on pourra utiliser la constante `Int.max_int` pour représenter la distance infinie

Q8. Modifier votre algorithme précédent pour écrire la fonction `dijkstra2:graphe->sommet->int array*int array` qui renvoie le tableau des distances minimales et le tableau des pères de chaque sommet dans l'arbre des plus courts chemins.

Q9. Ecrire la fonction `plus_courts_chemins:graphe->sommet->unit` qui affiche tous les plus courts chemins à partir d'un sommet donné. `Aller de Seville à Paris en 680 min : Seville Madrid Barcelone Paris ...` (★ : **appel professeur**)

Pour aller plus loin

Quelques variations autour de l'algorithme de Dijkstra ... y réfléchir sans forcément coder la solution.

Q10. Expliquez comment obtenir le plus court trajet entre deux villes données. Optimisation ?

Q11. On appelle **excentricité d'un sommet** la plus grande distance entre ce sommet et tous les autres sommets du graphe. Comment déterminer cette valeur ?

Q12. On appelle **centre du graphe**, le sommet d'excentricité minimale. Comment trouver ce sommet ? Cette excentricité est alors appelée **rayon du graphe**.

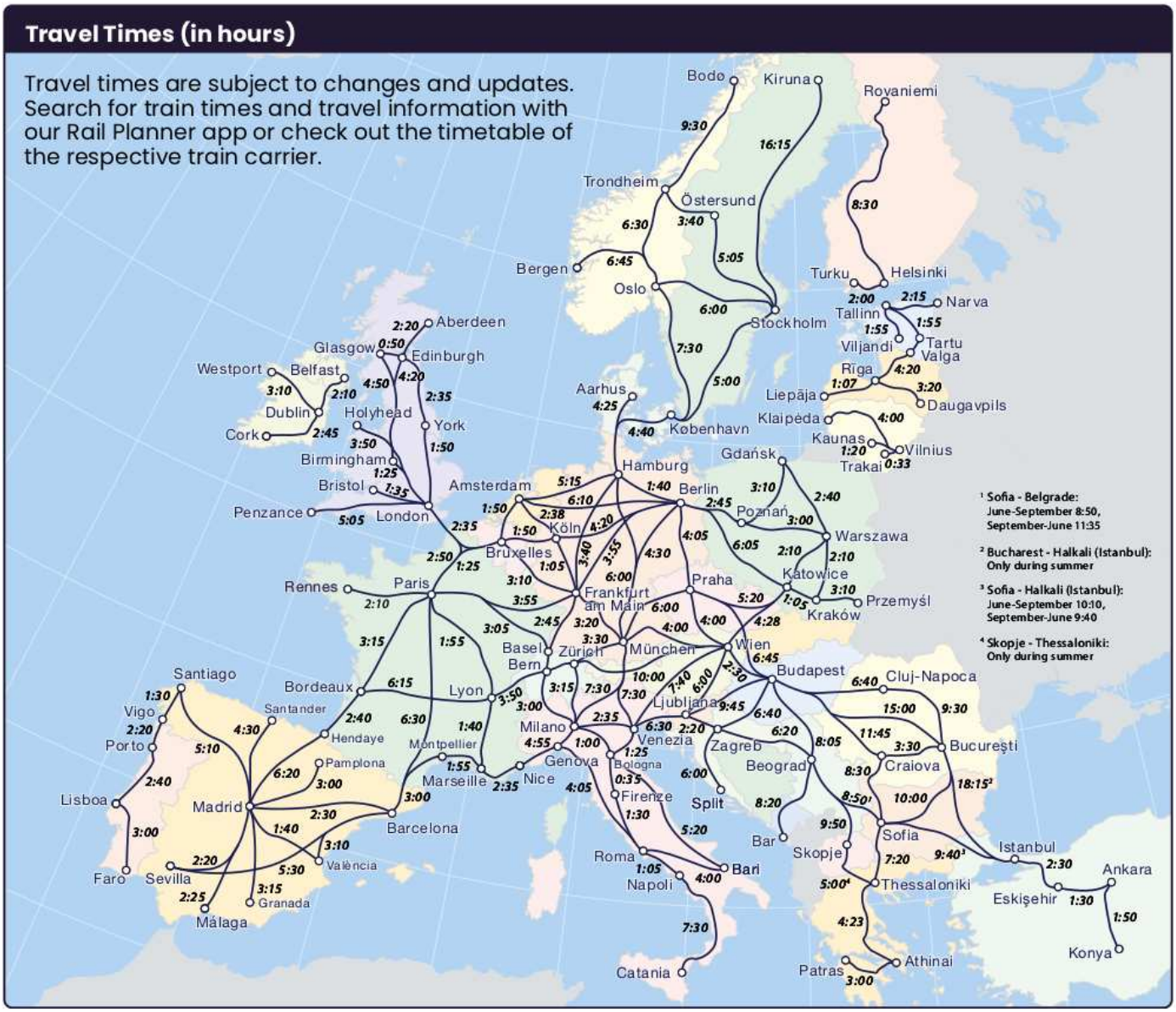
Q13. On appelle **diamètre du graphe**, la plus grande distance entre tout couple de sommet du graphe. Comment déterminer cette valeur ?

Q14. Comment retrouver ces différentes valeurs avec l'**algorithme de Floyd-Warshall** ?

Un plus gros graphe ...

Pour le premier à finir le TP ...

Ecrire les fichiers `villes2.txt` et `liaisons2.txt` correspondant au réseau ci-dessous, et tester vos algorithmes, et ainsi que ceux de vos camarades, avec ces fichiers.



Source : interrail.eu