Romain GOLDENCHTEIN
Raphaël CANAVATE
16 décembre 2023

[NF06] PROJET 2 : Tracé d'un réseau ferroviaire

LES TROIS FICHIERS ESSENTIELS SONT ***main.py*** , ***algo_prim.c*** ET ***dijktra.c***

TOUS LES FICHIERS CODE SONT **COMMENTÉS**.

Les fichiers dans le dossier *Avancement* sont juste des tests et autres.

Les fichiers dans le dossier *lib* sont des dépendances utilisées par les programmes principaux (txt, csv, ico ou python).

Les fichiers dans le dossier *src* sont des fichiers avec du code source.

Les fichiers dans le dossier *build* servent à l'exécution de *main.exe*.

POUR FAIRE MARCHER LE PROJET :

- Exécuter main.exe (si rien ne se passe ou *erreur:fichier_introuvable*, exécuter *algo_prim.c* une fois, *dijkstra.c* une fois, puis réessayer).
  (Si présence d'autres erreurs, essayer de modifier le chemin d'accès dans *main.py* , les 2 chemins dans *algo_prim.c* (et *kruskal.py*) et enfin celui dans *dijkstra.c*. Il faudra alors exécuter *main.py* au lieu de *main.exe*)
  (Il est aussi possible qu'il faille installer les bibliothèques Python requises dans main.py)
- Choisir, avec le bon numéro, quel algorithme est désiré.
- *carte.html* s'ouvre normalement dans une page web (sinon ouvrir manuellement).
- Une fenêtre Tkinter s'ouvre aussi.
- Dans la fenêtre Tkinter, sélectionner les villes voulues, cliquer sur "*Mettre à jour la carte*" et actualiser la page web.
  (À chaque mise à jour le temps d'exécution est écrit dans le terminal)
- Le tracé devrait être mis à jour (sinon il y a un problème avec les chemins d'accès).
- En cliquant sur un pin, il est possible de voir toutes les villes auxquelles sont reliées ce pin ainsi que les distances associées.
- En cliquant sur une ligne la distance associée devrait s'afficher.
- En cliquant n'importe où sur la carte les coordonnées devraient s'afficher.
- En cliquant sur "*Chemin entre 2 villes*" une nouvelle fenêtre Tkinter est créée où il est possible de voir le chemin entre 2 villes et la distance associée.
  (Cette fenêtre se ferme automatiquement lorsque la carte est mise à jour)
- Recommencer autant de fois que voulus !

Description des fichiers présents dans le projet :

- *main.py* : fichier principal, permet l'affichage de l'interface graphique mais aussi la communication avec les fichiers C.
- *main.exe* : exécutable de *main.py* (ne marchera plus si des modifications sont faites dans *main.py*)
- *algo_prim.c* : On fait tourner l'algorithme de **Prim** à partir des villes sélectionnées sur la fenêtre Tkinter.
- *kruskal.py* : Il est possible de faire tourner l'algorithme de **Kruskal** (en Python) au lieu de celui de Prim (en C). Le but était de voir si notre implémentation de l'algorithme de Prim était bonne.
  Effectivement, le graphique retourné sera le même (arbre de recouvrement minimale) mais la méthode pour le calculer différente (il est aussi possible de comparer les temps d'exécution entre Python et C).
- *dijkstra.c* : à partir du graph calculé par *algo_prim.c*, renvoie le chemin et la distance totale.
- *extraction.py* : outil utiliser pour extraire toutes les villes avec plus de 50 000 habitants du fichier ensemble.xlsx de l'INSEE (quelques autres villes ont été ajoutées).
- *tracer_graphe.py* : trace un graphique simple (avec *matplotlib*) avec une matrice en entrée.
- *mat.csv* : fichier utilisé pour enregistrer la matrice composée du poids (distance) de toutes les liaisons entres les villes.
- *villes_select.txt* : fichier utiliser pour enregistrer toutes les coordonnées des villes à relier qu'*algo_prim.c* (ou *kruskal.py*) utilise pour calculer la distance les séparant et remplir *mat.csv*.
- *villes.py* : Dictionnaire avec toutes les villes avec plus de 50 000 habitants et leurs coordonnées.
- *villes_americaines.py* : Dictionnaire avec toutes les villes des États-Unis avec plus de 10 000 habitants (et les villes connues de France) et leurs coordonnées.
  Ce fichier conséquent a surtout servi de tests pour comparer le temps d'exécution.
- *graphe.txt* : fichier utilisé pour enregistrer les villes reliées et la distance associée que *dijkstra.c* utilise pour calculer le chemin et la distance entre 2 villes du graphe

Sources :

- Cours sur les graphes : https://www-npa.lip6.fr/~blin/Enseignement_files/ALGR/ALGR_3_MST.pdf
- Algorithme de Prim (explications) : https://www.youtube.com/watch?v=cplfcGZmX7I
- Algorithme de Prim (structure) : https://openclassrooms.com/forum/sujet/algorithme-de-prim-70941
- Algorithme de Kruskal (explications) : https://www.techiedelight.com/fr/kruskals-algorithm-for-finding-minimum-spanning-tree/
- Subprocess: https://www.digitalocean.com/community/tutorials/how-to-use-subprocess-to-run-external-programs-in-python-3-fr
- Folium: https://python-visualization.github.io/folium/latest/user_guide/map.html
- Bibliothèque d'icônes : https://fontawesome.com/v4/icons/
- Tkinter : https://docs.python.org/fr/3/library/tkinter.html
- Fichier avec toutes les communes de France : https://www.insee.fr/fr/statistiques/6683035
- Formule Haversine : http://villemin.gerard.free.fr/aGeograp/Distance.htm
- Les manipulations de fichiers (et extraction) viennent du cours de NF06
- La fonction Dijkstra vient du cours (il a fallu la convertir en C néanmoins)