Rapport TP2 MTH6412B

Adrien Logut (1815142)

Retour sur TP1

Pour ce TP, nous avons repris le TP1 et nous avons pris en compte les retours qui ont été faits sur ce TP. La matrice d'adjacence a été rajoutée et est calculée à la fin de la création d'un graphe.

Choix d'implémentation

Kruskal

Dans ce TP, il nous est demandé de calculer l'arbre de recouvrement minimum d'un graphe connexe non-orienté. Pour cela, l'algorithme de Kruskal a été implémenté. Il se décompose en 2 grandes parties, l'*Union-Find* et l'algorithme en lui même.

Union-Find

Le principe d'Union-find est de représenter des ensembles, et que chaque élément d'un ensemble renvoie un représentant (avec la fonction *find*), et il est le même pour tous les éléments d'un même ensemble. Ces ensembles peuvent aussi être réunis (avec la fonction *union*), et tous les éléments des 2 ensembles auront dorénavent le même représentant.

Dans le cadre de ce TP, il a été choisi de représenter un represetant d'un ensemble comme étant le "parent" d'un ensemble. On forme donc un arbre. Cet arbre n'est pas forcement équilibré (piste d'amélioration possible)

La fonction *make_ set(sommet)* crée un ensemble singleton à partir d'un sommet. Son parent est lui même (donc son représentant).

La fonction *find(sommet)* renvoie le représentant de l'ensemble auquel ce sommet appartient. Cette fonction se charge de remonter l'arbre pour donner la racine de cet arbre et donc le représentant de cet ensemble.

La fonction *union(sommet$_1$, sommet$_2$)*, fusionne les ensembles dans lesquels sont sommet$_1$ et sommet$_2$. Pour cela, il set le parent du représentant de sommet$_1$ au parent du représentant de sommet$_2$.

Ces trois fonctions vont nous être utile pour Kruskal

Algorithme

L'algorithme en lui même est simple et se base sur l'*Union-Find*. Il renvoie la matrice d'adjacence correspondant à l'arbre de recouvrement minimum pour le graphe donné en entrée. Il retourne aussi le poids minimum.

Exemple du cours

Dans le fichier exempleCours.py, on retrouvre l'implémentation de l'exemple du cours pour l'algorithme de Kruskal. On peut l'exécuter comme cela.

Plot

Pour pouvoir tester les graphes et leur arbre de recouvrement minimal, nous avons réutilisé la fonction *plotGraph* de read_stsp.py pour l'ajouter à la classe Graph. Ainsi, *g.plot_graph()* affiche le graphe. Dans le cas où les noeuds n'ont pas de coordonnées données, on représente le graphe comme un polygone régulier à n côtés, n étant le nombre de noeuds.

De plus, si l'on rajoute en paramètre une matrice d'adjacence, cette fonction se charge d'afficher par dessus le graphe que représente cette matrice, et celà en noir. Cela permet de visualiser l'arbre de recouvrement minimum comme pour dans l'exemple du cours :

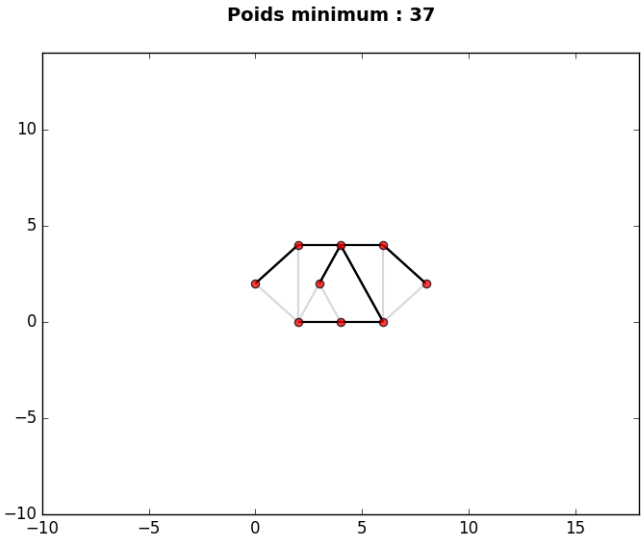

FIGURE 1 – Arbre de recouvrement minimum pour l'exemple du cours